UNIVERSIDADE D
COIMBRA

João Pedro Loureiro Ralho

Learning Single-View Plane Prediction from autonomous driving datasets

Dissertation supervised by Professor Doctor João Pedro de Almeida Barreto and submitted to the Electrical and Computer Engineering Department of the Faculty of Science and Technology of the University of Coimbra, in partial fulfillment of the requirements for the Degree of Master in Electrical and Computer Engineering, branch of Computers.

Internal Advisor: Prof. Dr. João Pedro de Almeida Barreto
External Advisor Dr. Michel Antunes

Coimbra, Setembro 2019

This work was developed in collaboration with:

University of Coimbra

UNIVERSIDADE D
COIMBRA

Department of Electrical and Computer Engineering

Abstract

Traditional 3D reconstruction using multiple images have some difficulties in dealing with scenes with little or repeated texture, slanted surfaces, variable illumination, and specularities. This problem was solved using planarity prior (PPR), which is quite common in man-made environments. Planar primitives are used for a more accurate, geometrically simple, and visually appealing reconstruction than a cloud of points. Single image depth estimation (SIDE) is a very appealing idea that has recently gained new prominence due to the emergence of learning methods and new ways to generate large accurate RGB-D datasets. This dissertation intends to extend the work developed in SIDE and work on single image piece-wise planar reconstruction (SI-PPR). Existing methods struggle in outputting accurate planar information from images because there are no large collections of accurate PPR data from real imagery. Therefore, this dissertation aims to propose a pipeline to efficiently generate large PPR datasets to re-train PPR estimation approaches. The pipeline is composed by three main stages, depth data generation using colmap, manual labeling of a small percentage of the images of the dataset, and automatic label and plane propagation to neighbouring views using a new strategy based on geometric constraints and random sampling. The pipeline created is able to efficiently and accurately generate PPR data from real images.

Abstracto

A reconstrução 3D tradicional usando múltiplas imagens apresenta algumas dificuldades em cenas com pouca ou repetida textura, superfícies inclinadas, iluminação variada e especularidades. Este problema foi resolvido através de geometria planar (PPR), bastante frequente em estruturas construidas pelo ser humano. As primitivas planares são usadas para obter uma reconstrução mais precisa, geometricamente simples, e visualmente mais apelativa que uma núvem de pontos. Estimação de profundidade através de uma única imagem (SIDE) é uma ideia bastante apelativa que recentemente ganhou novo destaque devido à emergência de métodos de aprendizagem e de novas formas de gerar grandes conjuntos de dados RGB-D precisos. No fundo, esta dissertação pretende extender o trabalho desenvolvido em SIDE para reconstrução 3D usando primitivas planares através de uma única imagem (SI-PPR). Os métodos existentes apresentam alguma dificuldade em gerar bons resultados porque não existem grandes coleções de dados PPR precisos de cenas reais. Como tal, o objetivo desta dissertação é propor um pipeline para gerar de forma eficiente grandes coleções de dados PPR para retreinar métodos de estimação PPR. O pipeline é composto por três partes, uma responsável por gerar informação sobre a profundidade numa imagem através do colmap, segmentação manual dos planos verificados na imagem, e uma propagação automática da segmentação realizada e dos parâmetros dos planos para as imagens vizinhas usando uma nova estratégia com base em restrições geométricas e amostragem aleatória. O pipeline criado é capaz de gerar dados PPR com eficiência e precisão a partir de imagens reais.

List of Figures

List of Tables

Contents

List of Acronyms

DL	Deep Learning
PPR	Piece-wise Planar Reconstruction
SI-PPR	Single Image Piece-wise Planar Reconstruction
SIDE	Single Image Depth Estimation
SfM	Structure-from-Motion
MVS	Multi-View Stereo
CNN	Convolutional Neural Network
RANSAC	RANdom SAmple Consensus
CRF	Conditional Random Field
MRF	Markov Random Field

1

Introduction

1.1 Motivation

1.1.1 Multi-view 3D reconstruction

Multi-view 3D reconstruction and Structure-from-Motion (SfM) are two well investigated topics in the computer vision literature [25, 14, 20]. Existing pipelines typically rely in matching points across consecutive views [19], and using geometric reasoning to recover the camera motion and reconstructing those points via triangulation [14, 20]. As described, the principle for recovering the scene structure from a set of images is simple. Nevertheless, and referring to Figure 1.1 and Figure 1.2, existing methods still have difficulties in handling situations of low or repetitive texture, variable illumination, surface slant and specularities [6, 25].

Figure 1.1: Textured scenes usually do not cause difficulties to current state-of-the-art algorithms.

Figure 1.2: Scenes that usually do cause difficulties to current state-of-the-art algorithms.

1.1.2 Single Image Depth Estimation (SIDE)

Reconstruction of 3D models and structures is usually accomplished through stereo vision, which consists in analyzing the photo-consistency or similarity between pixels of two different

images of the same scene, and then employing geometric constraints for inferring the 3D structure. However, the depth estimation using a single image has aroused some curiosity, specially with the emergence of learning methods.

Single Image Depth Estimation (SIDE) is an area of computer vision that is very important for the development of applications in the areas of 3D model reconstruction, robotic interaction, augmented reality and more. Traditionally, SIDE was performed taking into account variations in color intensities [15] or gradients or through geometric conclusions that can be drawn from focusing and blurring processes [10]. With the emergence of learning-based technologies such as Deep Learning (DL), traditional methods have been supersede. Deep learning methods applied to SIDE have better results than previously established SIDE methods, mainly because of the huge amount of existing datasets for training SIDE pipelines [31].

1.1.3 Pipeline for RGBD data generation to train deep learning algorithms

In the last few years, the advances in DL [13] motivated exploratory work in 3D computer vision towards recovering camera motion and/or scene depth using learned models instead of deterministic geometric reasoning. An important accomplishment of this effort is the multitude of solutions for Single Image Depth Estimation (SIDE) [18] that uses either supervised [16] or unsupervised [2] learning to perform 3D reconstruction from a single 2D image, which is something out of reach of classical geometric methods that usually require two or more views. Nevertheless, the available solutions for SIDE still have important limitations in terms of accuracy and generalization. As discussed in [18], the quality of 3D reconstruction of existing SIDE methods is still lacking accuracy for many real applications (e.g. robot path planning, augmented reality). This was experimentally analyzed in [18] using several state-of-the-art SIDE algorithms and the estimation of planar surfaces, where it was shown that the computed depth on those surfaces was inaccurate. Also, the depth edges tend to be over-smooth for most of the existing methods. Attempts to overcome limitations not only pass by improving the learning methodology but also by obtaining large quality datasets. The larger and more diverse the information, the better the final results of the algorithms and their generalization capability. Data for training SIDE approaches usually consists in a set of RGB images with its corresponding depth map obtained by RGB-D sensors (such as Kinect) and/or laser scanners (such as LiDAR). These devices, however, have some problems. Kinect is a great and inexpensive RGBD imaging device but works best for indoor scenes. LiDAR is a laser sensor that can be used to obtain RGBD data in outdoors scenes but strong sunlight can cause interference, adding considerable noise to the depth information. Recently an alternative to these types of equipment for obtaining training data has emerged. Li and Snavely [31] proposed a pipeline, called MegaDepth, which combine multi-view stereo (MVS) and structure-from-motion (SFM), followed by post processing methods to generate datasets to train SIDE networks. This thesis is largely inspired by Megadepth [31]. The objective is to create a pipeline that not only generates RGB-D data, but goes one step further and also compute piecewise planar information about the scene.

1.1.4 Multiview 3D piece-wise planar reconstruction

Since man-made environments are dominated by planar surfaces, many authors [12, 4, 1] have used the planarity assumption to overcome the issues mentioned in the Section 1.1.1. The idea is to replace the point primitives in traditional 3D reconstruction and SfM algorithms [25, 14, 20] by plane primitives, and, thus, obtain a Piecewise-Planar Reconstruction (PPR) of the scene that is more accurate, geometrically simpler and visually more compelling than sparse Point Clouds (PCs).

The ISR-UC made important contributions to this topic by developing pipelines for both stereo [4, 1] and monocular sequences [3]. The general idea is to pose plane hypotheses in the scene using either geometric or learning based frameworks and then using optimization strategies to assign, to each pixel in the image, one of those plane labels (the non-plane label is usually assigned for handling non-planar structures).

1.1.5 Single image piece-wise planar reconstruction (SI-PPR)

Single image piece-wise planar 3D reconstruction takes advantage of planarity prior to simultaneously detect planes and recover 3D plane parameters from a single image. There are no significant improvements of PPR based Deep Learning (DL) when compared to traditional PPR approaches based on geometric computer vision. Mainly because there are no large enough databases for PPR with accurate labeling and reconstruction of the planes in the scene. Also, most of the existing datasets for PPR were either generated synthetically, do not have the plane labels [27] or were obtained from RGB-D data with time consuming manual labeling of objects and structures, which were in a second stage considered planar or not planar [9]. The lack of large, accurate and labeled datasets is one of the major difficulties for accurately training DL SI-PPR approaches. This thesis tackles this problem by presenting a pipeline to generate PPR data in a semi-automatic manner that is efficient and robust.

1.2 Contributions

1.2.1 Semi-supervised pipeline for generating large PPR datasets of real scenes

A new semi-supervised pipeline is proposed to generate large datasets of real scenes for PPR. This pipeline includes calibration of intrinsic parameters and provides for each image a planar segmentation mask, the parameters of the planes identified in the image, and depth and normal maps of an image. Will be generated a large PPR data of the streets of Coimbra.

1.2.2 Training of a recent DL based PPR approach using our own dataset generated in a semi-supervised manner

The results show that the real data generated by the pipeline created is sufficiently accurate to effectively re-train existing deep learning PPR networks.

1.3 Organization

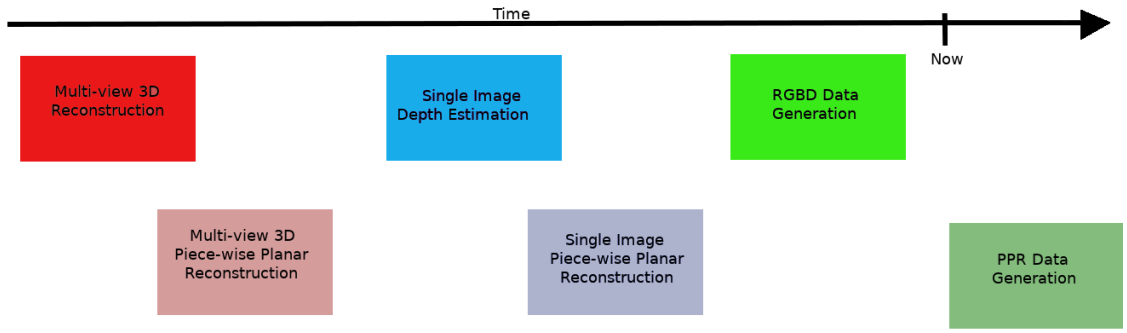

Figure 1.3: Visually representation of the progress of the state-of-the-art in 3D reconstruction. This thesis aims to do the PPR data generation.

This dissertation starts by introducing concepts visually represented in the figure 1.3. It starts by introducing some works of the literature regarding multi-view 3D reconstruction, single image depth estimation, and a pipeline to generate a large quantity of accurate RGBD data. Then it is explained how these methods are improved taking into account planarity prior. The problem is that there isn't yet a way to generate a large quantity of accurate PPR data to train PPR networks. This dissertation brings a solution to tackle this problem.

In Chapter 2, it will be introduced: how traditionally multi-view 3D reconstruction is done and the various methods involved, how single image depth estimation is done through geometric and learning methods, and a recent method for generating large amounts of RGBD data for training SIDE networks. Then, it will be talked about the advantages of doing multi-view 3D piece-wise planar reconstruction and how it is currently performed, and will be talked about two methods for single image piece-wise planar reconstruction and present some of the results obtained by them.

In Chapter 3, it will be discussed about the first part of the pipeline created to generate large quantities of accurate PPR data to train networks. The Colmap pipeline will be introduced, what can be done with it, and how it is used in the pipeline to generate the initial data needed to feed the algorithm that generates the PPR data.

In Chapter 4, it will be explained the three main parts of the algorithm and how the user can interact with it. The first part that receives the output from Colmap, the second part that ask the user to manually label the planes of an image and the last part that propagates the manual label

made to the other neighboring images of the dataset. It will be also discussed the dataset created with the algorithm and shown the results.

In Chapter 5, it will be explained how the dataset for PPR generated was used to train a SI-PPR algorithm, the metrics used to evaluate the results obtained and a comparison between the ground-truth (the PPR data generated), the results from the SI-PPR algorithm before re-training and after the re-training.

The Chapter 6 is dedicated to the conclusions made about the new way to generate large quantity of precise PPR data and how is a good solution to tackle the problem in obtaining this kind of information to train deep learning networks.

2

Literature review

2.1 Multi-view 3D reconstruction

Traditionally, 3D reconstruction is done using Multi-view Stereo Reconstruction Algorithms, using various methods to achieve this [25], such has : scene representation, photo consistency, model visibility, shape priors, the reconstruction algorithm, and initialization requirements.

In the representation of the scene is tried to explain the scene geometrically through simple shapes such as voxels, polygons or depthmaps. In the photo consistency, references are sought for a particular part of the image or feature, from image to image, establishing a relationship between them. For visibility of the model, is taken into account which images to use for photo consistency. In shape prior, certain characteristics are applied to the reconstruction in zones where photo consistency fails (such as poorly textured zones). The reconstruction algorithm can be categorized into four classes that differentiate according to the technique used for the purpose. Some algorithms calculate a cost function on a 3D volume and extract a surface from the model. Others iteratively add or remove inconsistent voxels from an original volume, trying to optimize an energy function. There are still algorithms that attempt to obtain and force consistency in consecutive image depthmaps to form a consistent 3D model. Still, others extract and match a set of features found in consecutive images followed by a problem of fitting the surfaces on those features. The geometric information needed as input for the algorithms are the initialization requeriments.

2.2 Single image depth estimation (SIDE)

Single Image Depth Estimation (SIDE) is an important problem that has benefited works in the areas of 3D reconstruction, robotic interaction, augmented reality and more [31]. Good results can be obtained through stereo methods, where it is necessary to use two or more cameras. Still, SIDE offers comparative advantages to the traditional methods to obtain depth (MVS and SFM) like the reduced need for hardware, less data needed, and no need for intrinsic calibration.

There are several methods for estimating depth through a single image. These methods can be geometric such as shape from shading and shape from defocus, or learning methods. In the

case of shading and shape, intensity or color gradient is taken into account to draw geometric conclusions about the structure of objects [15]. In the case of shape from defocus, the focus and blur behaviour is explored in order to extract geometric clues about the object or structure in question [10]. Learning methods try to teach machines to see the world the same way humans do by transforming data into abstract representations.

A Convolutional Neural Network (CNN) is a learning algorithm that receives an image as input, applies a weight/importance to the various elements in the image and ultimately differentiates them. This is possible through the use of filters that traverse the image by performing a convolution operation, and the use of reducing processes that decrease the spatial size of the convolution result by extracting dominant characteristics. By adapting the entire set of filters and pooling layers it is possible to teach a network how to estimate depth through a single image [21]. Eigen et al [7] created a two-component algorithm that first estimates the depth of the scene on a global scale and finally refines the estimation locally. Liu et al [11] combined CNN with Conditional Random Fields (CRF). CRF is a probabilistic model used to encode known relations between observations and construct consistent interpretations. Typically used in computer vision for object recognition and image segmentation. In the algorithm, CRF is used to establish relationships between neighboring super pixels.

2.3 RGBD data generation with Megadepth

Most RGBD datasets are generated using equipment such as Kinect and LIDAR. However, Kinect is a device that works well for exclusively indoor scenes, and LIDAR, besides being expensive equipment, can apply depth map noise in bright sunlight. As such, Megadepth proposes the use of a stereo and structure-from-motion multi-view pipeline called Colmap [22, 24, 23] followed by a set of cleaning processes to generate RGBD training data.

Colmap [22, 24, 23] is a Structure-from-Motion and Multi View Stereo pipeline that allows the user to perform 3D reconstructions from a set of images. With this algorithm, it is possible to obtain a model with the sparse reconstruction of the scene. The reconstruction will give information about the cameras, images, and points used in the model. Regarding the cameras, acquired information about the camera model used, the image size obtained by the camera (width and height) and the associated intrinsic parameters. In the case of images, obtained the information associated with the projection of the image from the world coordinate system to the camera coordinate system, that is, extrinsic/pose parameters. For the points used in the model, is obtained information about the coordinates of the points, their RGB value, error and also the index of the respective point in the image file. The algorithm also outputs a dense reconstruction model with depth and normal maps (with geometric and photometric information), a point cloud representative of the model reconstructed by merging these maps, a mesh of the model, a folder with undistorted images, and consistency graphics that define, for all pixels, the corresponding pixel which is consistent within the source image.

The depth map that comes from Colmap sometimes has many outliers that have negative effects on depth estimation networks. Given that Colmap acquires image depth iteratively, Megadepth has created a system that, in each iteration, compares the depth in each pixel with the previous iteration and stores the smallest value. In the end, is applied a median filter to the depth map.

Another process used for better results is the use of a semantic segmentation filter to clean out objects and zones that are not important.

Thus Megadepth can generate accurate data through collections of photographs taken from the internet, thus creating an alternative to obtaining large amounts of RGBD data for training SIDE networks.

2.4 Multi-view 3D piecewise planar reconstruction (PPR)

The goal of Multi-view stereo is to completely reconstruct a 3D model of an object through a set of images taken from different points of view. The traditional methods have difficulties dealing with poor texture, brightness variation, light reflection on certain surfaces and high slope surfaces. These difficulties can be overcome by exploiting the fact that most scenes created by humans regularly have planar characteristics. Taking advantage of planar geometry, it is possible to build geometrically simpler 3D models and less complex in terms of rendering, storage, and transmission, compared to the conventional point cloud models.

Generally speaking, Gallup et al [8] through an image sequence, intrinsic and extrinsic calibration and a dense depth map, obtain a set of planar and non-planar depth maps used to generate a polygon mesh representative of the scene. For this purpose, through the depth maps fed as input, planes are estimated using a method called RANSAC. RANSAC is an iterative method for estimating parameters of a mathematical model through a data set that contains significant differences (outliers). For each pixel of the depthmap, an MRF problem is applied to match the pixel to one of the planes estimated by RANSAC. There is also a correspondence to the possibility that the pixel does not belong to any plane. The MRF is a set of random variables, where their future states depend only on the present state, a property that is described by an undirected graph. Gallup et al still use a classifier to more successfully distinguish what is plane and non-plane and ensure consistent 3D reconstruction between images by merging of the initial estimated planes over overlapping views.

Raposo et al [4] created a pipeline that through a sequence of images is able to obtain the motion of the camera and the 3D planes of the scene. The pipeline begins by detecting planes of the scene followed by calculation that gives the relative pose between consecutive images using a RANSAC. To merge the various plane estimates over the stereo pair, a PEaRL framework [17] is used to perform joint refinement of motion and structure. A PEaRL framework efficiently exploits the continuity of labels in energy-based geometric model fitting.

2.5 Single image piece-wise planar reconstruction

The methods described in the previous section are geometric methods for PPR. However, with the emergence of deep learning, algorithms were appearing to learn how to reconstruct 3D planes through planar primitives requiring only an RGB image as input. Through these planes is easily constructed a planar piece-wise 3D model of the scene.

2.5.1 Planenet

PlaneNet [9] is a work made by *Liu et al* [9] that do a PPR from a single RGB image. Plane parameters and their segmentation maps are needed to do a depthmap. *Liu et al* solve this problem by letting the network learn to directly produce a set of plane parameters, a probabilistic plane segmentation mask and predict a depthmap at non-planar surfaces (the three outputs of this algorithm).

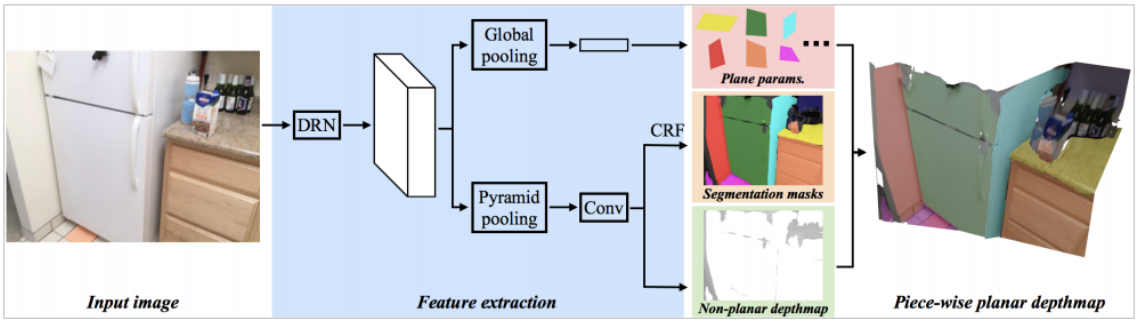

Figure 2.1: PlaneNet [9] architecture

The algorithm architecture is divided into three branches from where is obtained the different outputs. PlaneNet is built upon Dilated Residual Networks (DNR) [28] followed by Global Average Pooling [28] (first branch) from where we obtain the plane parameters. Dilated Residual Networks are based in dilated convolutions that, in simple terms, are convolutions with wider kernels.

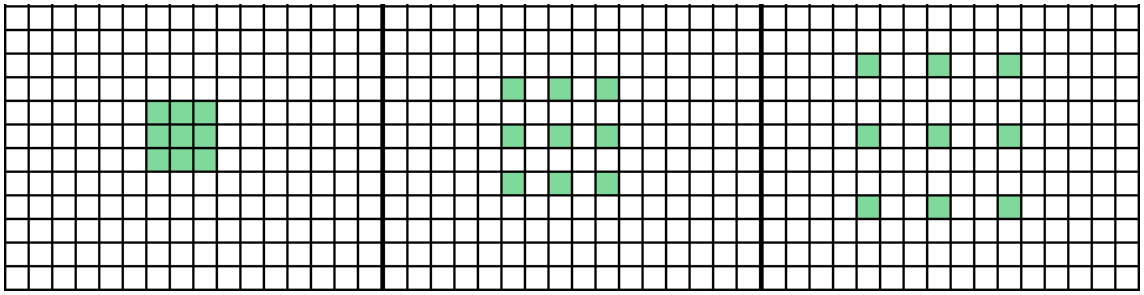

Figure 2.2: Kernel used in convolution with different dilation values. From the right to the left can be seen dilation=1, dilation=2 and dilation=3.

This type of network is used to obtain larger receptive fields in the upper layers to compensate for the reduction in the receptive field due to downsampling. Global Average Pooling is an operation that averages each feature map of the previous layer, thus reducing a large number of parameters used for training which consequently affects the over-fitting tendency. From the output of Dilated Residual Network, is fed a Pyramid Pooling Module [29] and then a Dense Conditional Random Field (DCRF) [30] (after some convolutional layers) to get the segmentation masks (second branch). The Pyramid Pooling Module acquires different representative sub regions of a feature map, upsamples and concatenates to form a final feature map with global and local contextual information. Conditional Random Field is a statistical modeling method used to encode known relationships between observations and build consistent interpretations. The last branch is built like the second branch but without the Dense Conditional Random Field (third branch). 50 000 images are used from ScanNet (a RGBD video database) to train the network and more 1 000 for testing purposes. Some problems can be found in planes where the light changes along the surface of the plane, and when there is little to no difference in the texture in two close planes. Some results can be seen in Figure 2.3.

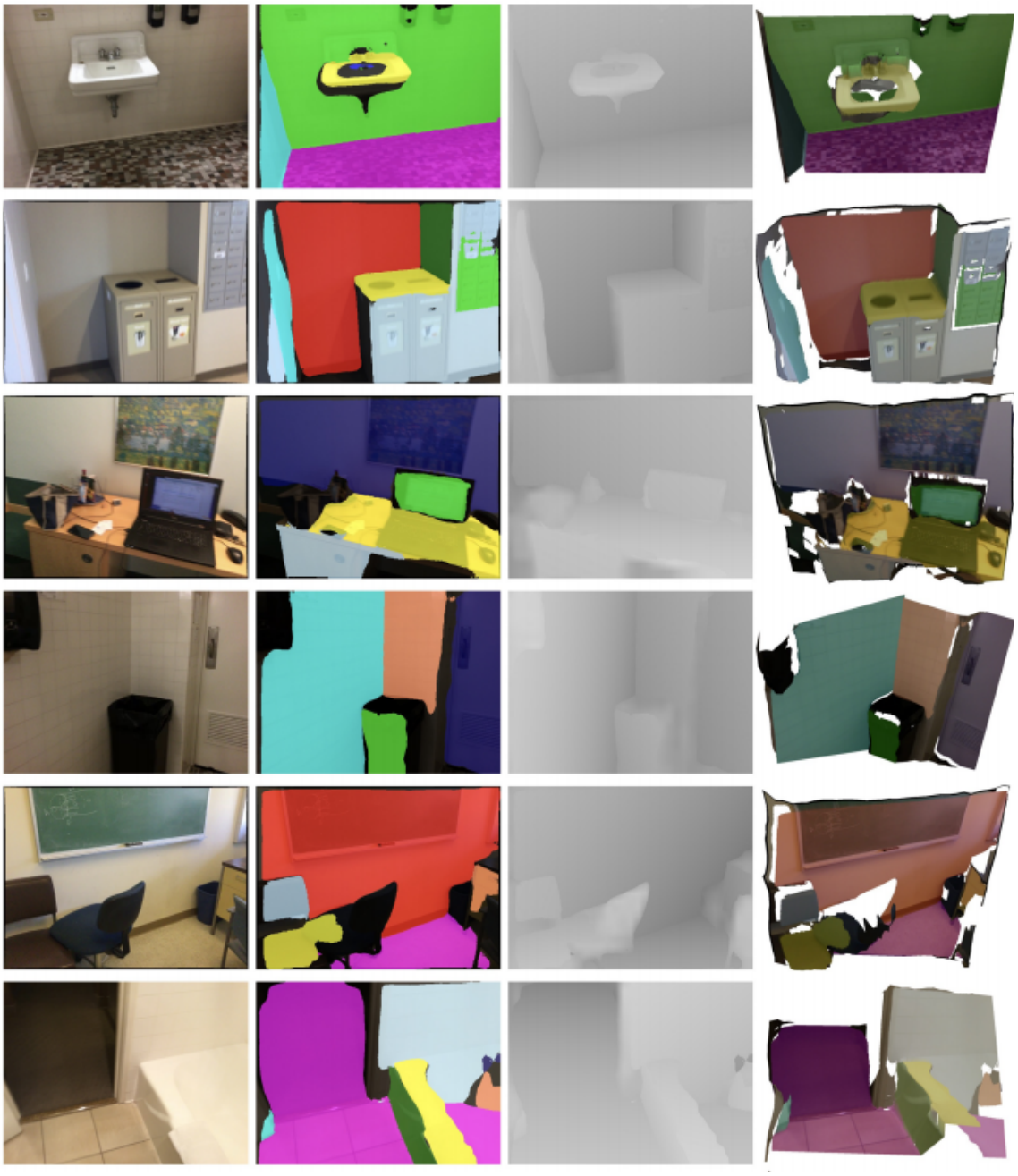

Figure 2.3: PlaneNet [9] results in ScanNet images. The left image is the input used followed by the plane segmentation, depth map reconstruction and 3D rendering of the depthmap.

2.5.2 R3PSICNN

PPR can be obtained with different methods that can be geometry based or appearance based. Geometry based methods take advantage of parallel lines (where we try to determine the vanishing points and determine the 3D plane orientation) and orthogonal lines (more particularly the junction between to lines, if these junctions form a 90º degrees angle then we can infer the existence of rectangular structures). Apperance based methods infer geometric properties of an image from its

appearance (by certain features like texture, shape and color). This paper [27] studies the problem of recovering 3D planar surfaces from a single image using apperance based methods. This work simultaneously predicts a plane segmentation map and the parameters of the 3D planes. The paper describes how they obtained a 3D point from a 2D point (using the image, the corresponding depthmap and the intrinsic matrix of the camera used to take the picture):

$$Q = D_i(q) \cdot K^{-1}q$$

Where the 2D point is represented by q, its depth value by $D_i(q)$, the intrinsic parameters by K and the 3D point by Q. Having the 3D point and the equation of a plane it is possible to see if the point belongs to that same plane because:

$$n^T Q = 1$$

Where n corresponds to a plane in R^3. Then If the 3D point is on a specific plane then we have the following:

$$(n_i^j)^T \cdot \lambda \cdot K^{-1}q = 1$$

$$\lambda = \frac{1}{(n_i^j)^T \cdot K^{-1}q}$$

Where λ is the depth at q constrained by j-th plane in the image i, n_i^j. This knowledge is used to train a network to output a per-pixel probability map and the plane parameteres by minimizing their loss function:

$$\sum_{i=1}^{n}\sum_{j=1}^{m}(\sum_{q} S_i^j(q) \cdot |(n_i^j)^T Q - 1|) + \alpha \sum_{i=1}^{n} Reg(S_i)$$

The first part of the equation treats the 3D plane recovery problem as a depth prediction problem by comparing the depth induced by the predicted plane λ with the ground-truth $D_i(q)$:

$$|(n_i^j)^T Q - 1| = |(n_i^j)^T D_i(q) \cdot K^{-1}q - 1| = |\frac{D_i(q)}{\lambda} - 1|$$

The second part of the loss function is a regularization term that helps distinguish what is or is not a plane in the image. Letting $z(q) = 1$ if pixel belongs to a planar surface and $z(q) = 0$ otherwise, we have:

$$P_{plane}(q) = \sum_{j=1}^{m} S_i^j(q)$$

$$Reg(S_i) = \sum_{q} -z(q) \cdot log(P_{plane}(q) - (1 - z(q)) \cdot log(1 - P_{plane}(q))$$

Here, P_{plane} is the sum of probabilities of pixel q being assigned to each plane. The α before the regularization term $Reg(S_i)$ prevents the classification of all pixels as non-planar.

So, to obtain the outputs, they build the network in such a way that the architecture is divided into two prediction branches.

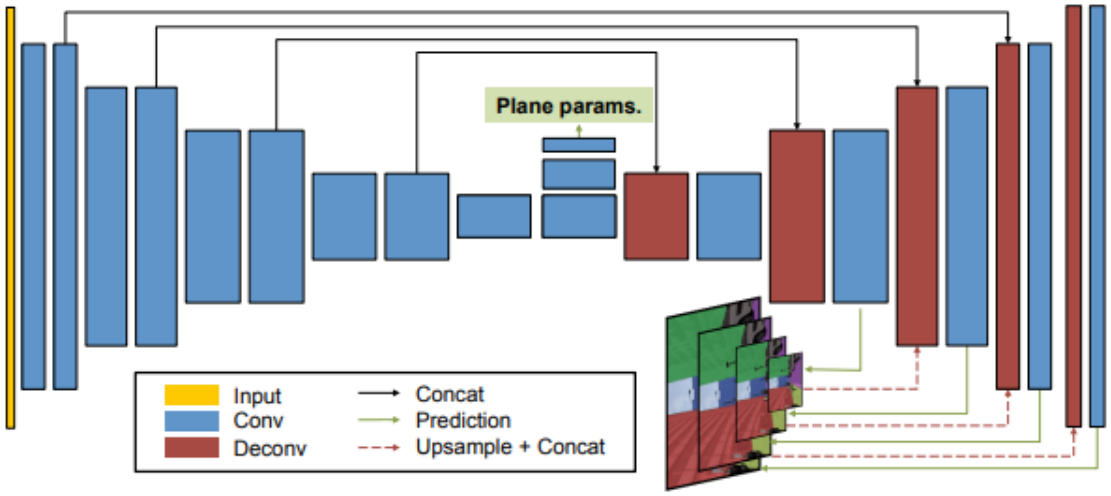

Figure 2.4: R3PSICNN [27] architecture

There is a branch dedicated to generating a plane segmentation map using an encoder-decoder design with skip connections and multi-scale side predictions using ReLU activation functions for all layers except for the last ones where is used softmax instead.The encoder will gradually downsample the image through convolutions, producing high-level feature maps. The decoder will upsample features maps through deconvolution layers. Activation functions are responsible for mapping values that come out of layers between certain values defined by the function. In the case of the ReLU function, the negative values are all converted to zero and the positive values keep their value.

The other branch is dedicated to acquiring plane parameters. This branch consists of stride-2 convolutional layers followed, in the end, by a global average pooling layer to aggregate predictions across all spatial locations. As activation functions, is used ReLU except in the last layer where none is necessary.

Two datasets are used: SYNTHIA and Cityscapes. SYNTHIA is a set photo-realistic synthetic images of urban scenes with the corresponding depth maps noise free. Cityscapes is a real street-view video sequence. From SYNTHIA, 8 000 images were randomly selected to train the network

100 for testing. From Cityscapes, 100 were taken for testing purposes and the 3 475 for training. In the end, can be seen good results from their images but some with some failures like the separation of one plane into two or merge of multiple planes into one. It can also be noted some errors when multiple planes are at great distances. Some results can be seen in Figure 2.5 and Figure 2.6.

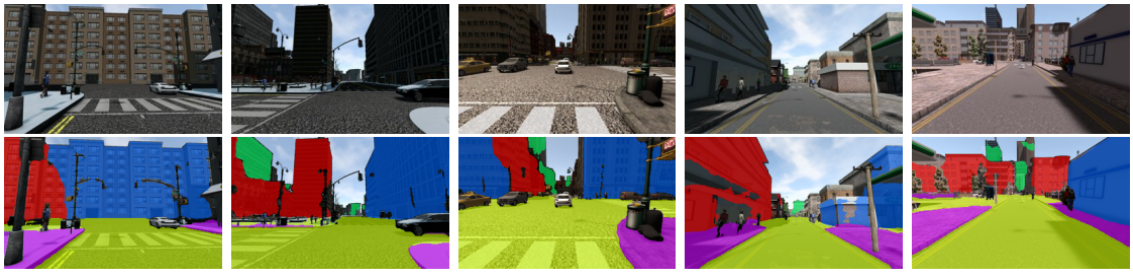

Figure 2.5: R3PSICNN [27] bad results in SYNTHIA images.

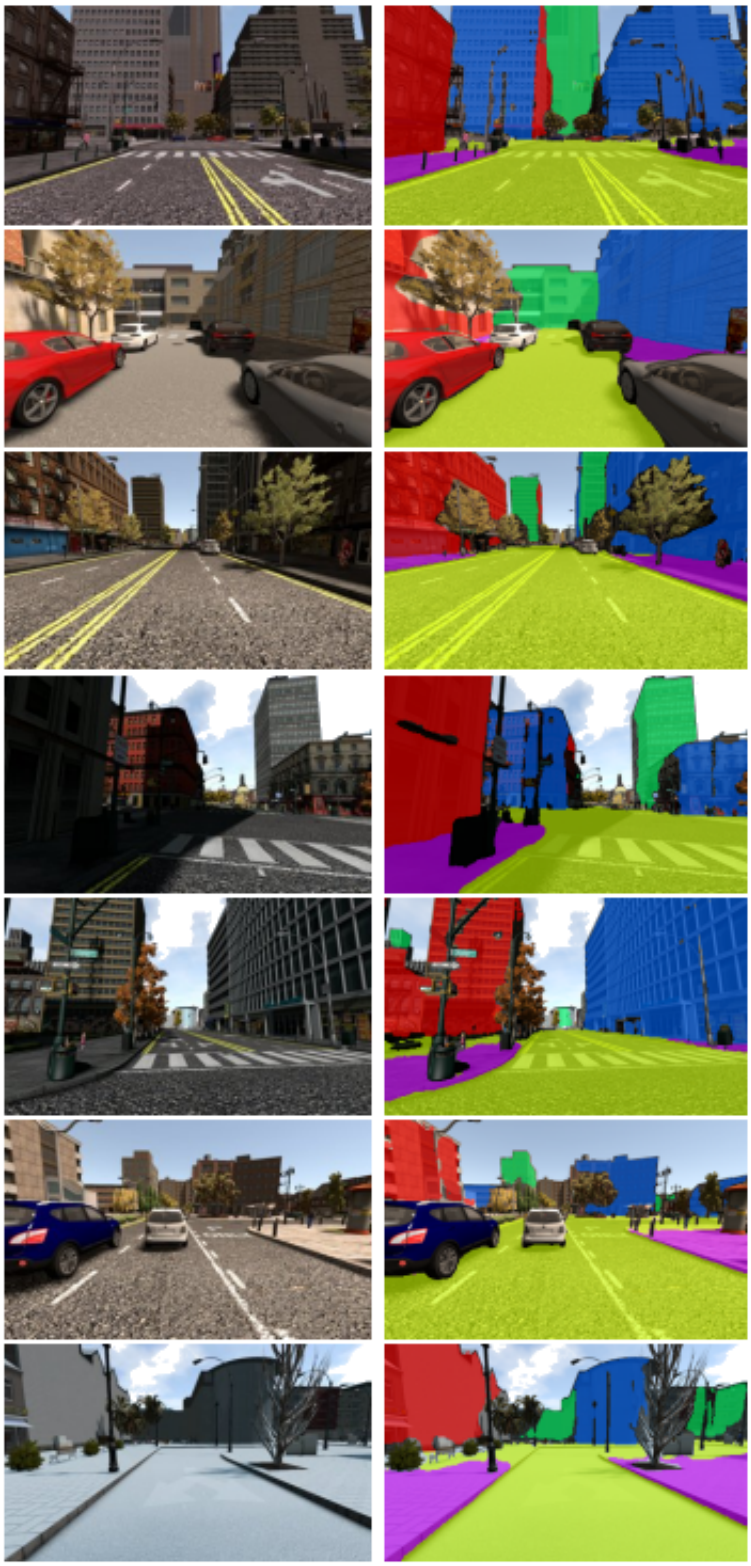

Figure 2.6: R3PSICNN [27] good results in SYNTHIA images. The left image is the input used and the right image is the segmentation made by the algorithm.

3

Structure-from-Motion and Multi-View pipeline

As said in the section 1.1.3, good and accurate RGBD data is difficult to obtain. The main reason lies in the hardware limitations and the difficulty of operating them. In this chapter will be described the structure-from-motion and multi-view stereo pipeline used in Megadepth [31] and how this pipeline was used to generate input data to the PPR data generation algorithm.

3.1 Colmap pipeline

Colmap [22, 24, 23] is a Structure-from-Motion (SfM) and Multi-View Stereo (MVS) pipeline that allows the user to perform 3D reconstructions through a set of images. This is made in three different steps.

In the first stage occurs the detection and extraction of features. Here are found features in the image that, as a rule, are invariant to radiometric and geometric changes. Thus, it is possible to observe the same features in multiple images. In this step is given the possibility to choose between using the intrinsic parameters of the camera model used or, if these are unknown, let Colmap automatically extract the focal length information using the Exchangeable Image File Format (EXIF, specification of digital cameras on the conditions of the image capture, stored as metadata in the image taken). Still, in this step, it is possible to detect and extract new features from images or import known features stored in text files. If is needed to extract new features, some options can be changed such as maximum image size, the maximum number of features that can be detected in the image, use of GPU and which GPU to use (to speed up the process), among others.

In the second stage there is the correspondence of the characteristics, detected and extracted from the previous one, and their geometric verification. At this stage, SfM looks for images where it is possible to observe the same characteristics. Here, image B is searched for the feature most similar to the feature of an image A, with the help of a comparison metric. In the end, an output corresponding to a set of possible overlapping images is obtained. These sets of images are geometrically checked through transformations that attempt to map features between images using projective geometry. Colmap, to apply this step, gives different options useful for different situa-

tions. For feature matching, its possible to choose between an exhaustive, sequential, vocabulary tree, spatial, transitive, or custom match:

- Exhaustive matching competes for small datasets, with just a few hundred images. For this kind of datasets, this match gets the best reconstruction.
- Sequential matching is ideal for video sequences, where there is a logical spatial sequence between each image, and it is possible to visually detect characteristics between consecutive images. In this mode, can be enabled loop detection and also be modified some parameters such as the number of images to overlap, the frequency (in the number of images) to which an image is matched to the images most similar to it, maximum features, among others.
- Vocabulary tree matching is recommended for a large collection of images (with thousands of images). Here, each image correlates with the nearest neighbor images using a vocabulary tree. The number of neighboring images to consider and the maximum number of features can be changed.
- Spatial matching can be helpful if there is accurate GPS information about the location where the photograph was taken. This information can also be taken from the EXIF. Here, each image is matched by each neighboring image spatially. Certain settings can be changed for better reconstruction results. It is possible to take into account the maximum acceptable distance between each image, the maximum number of neighboring images to consider and even, if the user wishes, ignore the zz axis.
- Transitive matching uses transitive relationships to improve matching between images. For example, if one of two correlated images has anything to do with a third image, colmap attempts to match all three.
- Custom Matching lets you specify specific image pairs to match.

In last step is made the sparse and dense reconstruction of the structure. All information obtained from the previous steps is loaded and the scene's incremental reconstruction is started. Then depth and normals of each pixel of each scene image can be retrieved, and by merging this new output it is possible to obtain a dense point cloud. The point cloud can then be used to estimate a dense surface.

3.2 Steps taken in Colmap

Colmap [22, 24, 23] must be fed a sequence of images so that it can perform a sparse and dense reconstruction, providing as output the cameras used in the reconstruction, the images, the 3D points, the depth and normal maps of the images.

To do this, the user must open the Colmap graphical interface and create a new project. Here the user must let Colmap know where the dataset is located. The images will be loaded and then the features can be extracted:

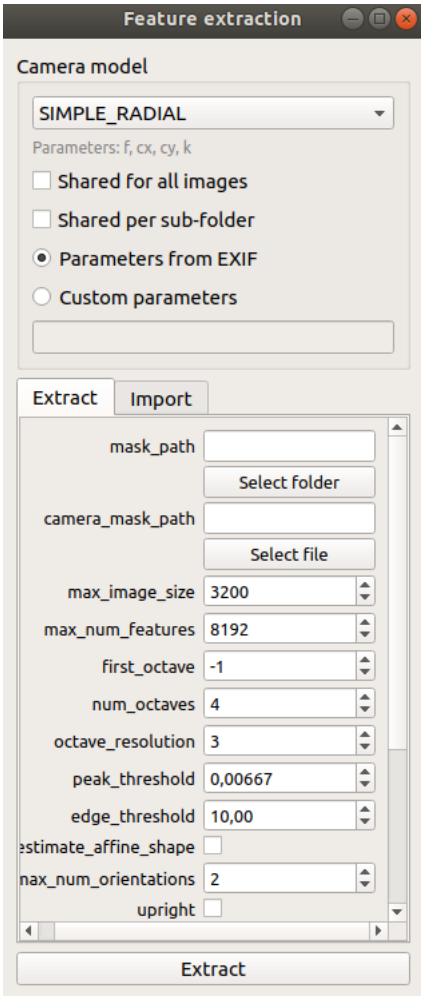

Figure 3.1: Colmap graphical interface. Feature extration window.

In this dissertation was used several datasets created with images from different areas of the city of Coimbra (Portugal). The datasets were obtained through stereo cameras mounted on the top of a car with well known intrinsics parameters and model. In the settings for feature extraction, the camera model chosen was $SIMPLE\_RADIAL$, was checked the box to apply the intrinsic parameters to all images from the datasets, and put the intrinsic parameters in the $Custom\ parameters$. The rest of the options stayed the same. After all the changes, the features were extracted.

The next step is to match the features of the different images. Colmap, for each image, will take the extracted features and establish relationships between the different images of the dataset. The following window will appear:

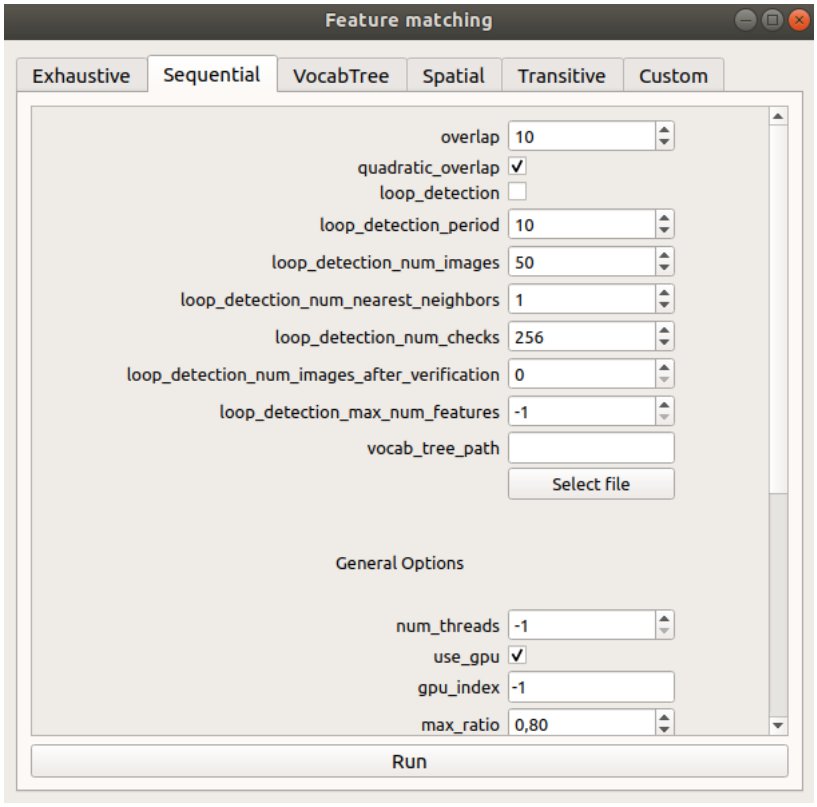

Figure 3.2: Colmap graphical interface. Feature matching window.

From the various scenarios available, was chosen the scenario $Sequential$. In the settings, in $overlap$, the value was changed to 30 and was checked the box that allows Colmap to detect a loop, $loop\_detection$, in the datasets where the car in which the photos were taken came back to the starting point. For the $vocab\_tree\_path$ option, is needed to download the vocab_tree available from the Colmap website. The rest of the options stayed on default. After some minutes, the matching is complete and the model is ready to start sparse reconstruction. The reconstruction can be seen in real time.

The last step is to do dense reconstruction in order to obtain the dense and normal maps for each image of the dataset.

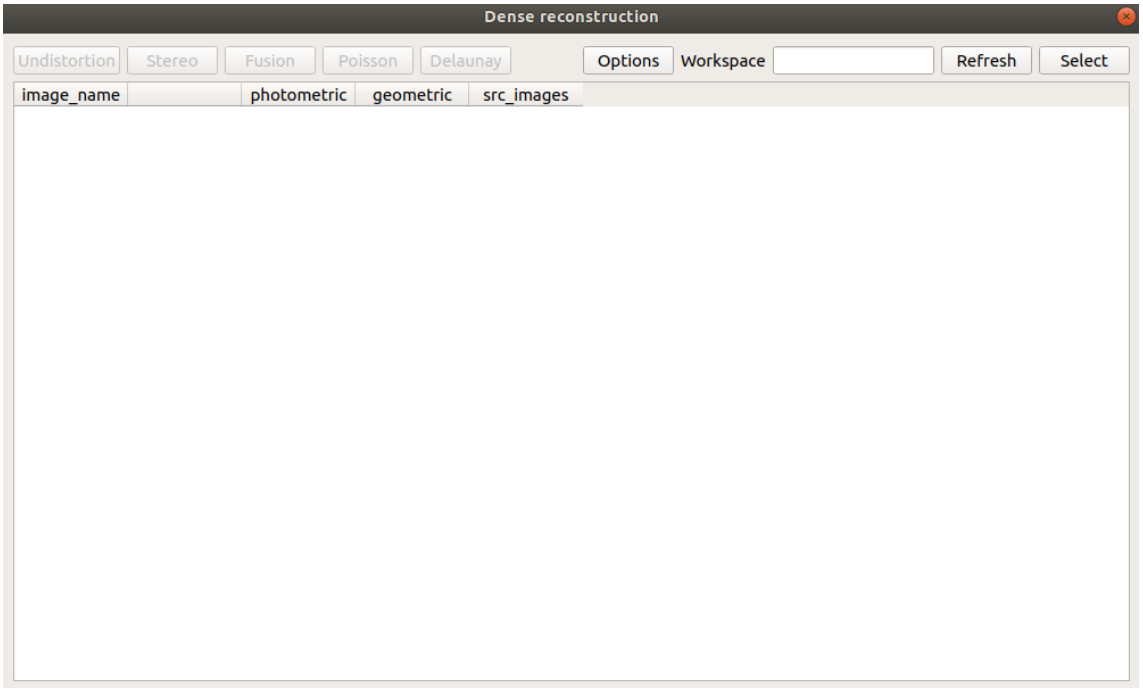

Figure 3.3: Colmap graphical interface. Dense reconstruction window.

The user can undistort the images, get the depth and normal maps for each image, merge the referred maps to create a point cloud and finally meshing the cloud points (using the Poisson or Delaunay reconstruction). This must be done sequentially. The depth and normal maps cannot be generated without first removing distortion from the images. For the propagation algorithm to run, only depth and normal maps are required from the dense reconstruction. The depth and normal maps are saved in the *.bin* format folder but can be read with a script provided by the authors.

Before closing the Colmap, its necessary to export the model as a text file. This means that the information related to the cameras, the images and the 3D points will all be in text format.

4

Generation of PPR data for learning single image plane prediction

Any kind of deep learning algorithm needs a considerable amount of data to train its models. The quantity and quality of data fed to the neural network for training play a key role in making the algorithm accurate and good to generalize. The 3D reconstruction networks that take advantage of planar primitives, are no exception. PlaneNet [9], a deep learning based PPR algorithm, requires plane segmentation, its parameters, and the depthmap of the image. All of these inputs are important for minimizing the loss functions and for good inference results. In R3PSICNN [27], for network training, it must be fed a depthmap of the image and a binary segmentation of what is planar and not planar. These inputs are required to minimize the loss function referred section 2.5.2. To work with the learning based PPR algorithms, its needed information about the plane parameters, where to find them in the images and image depth maps. However, this type of information is difficult to obtain. As noted Section 1.1.3, RGBD information is relatively difficult to acquire due to hardware limitations and difficulty operating them. Very accurate plane information for re-training SI-PPR is usually obtained through cumbersome and time consuming manual labeling of planes found in an image, followed by a whole problem of plane fitting to obtain their parameters.

Inspired by the difficulty in obtaining training data for PPR networks and the work of Megadepth [31], was developed an algorithm for generating a large quantity of accurate PPR data.

4.1 Algorithm for generating PPR data

Figure 4.1: The 3 main parts of the algorithm.

The algorithm is divided into 3 parts. The first part is responsible for reading the Colmap ex-

ported model and organizing the information into specific structures in addition to arranging the images in sequential order. The information is then stored in a mat file that will be read by the second part of the algorithm.

In the second part, depth and normal maps are read and placed in specific variables, intrinsic parameters are put in a matrix for future calculations, and arrays and matrixes are initialized. When the algorithm starts, one image from the dataset opens and the user is asked to manually label planes that are visible.

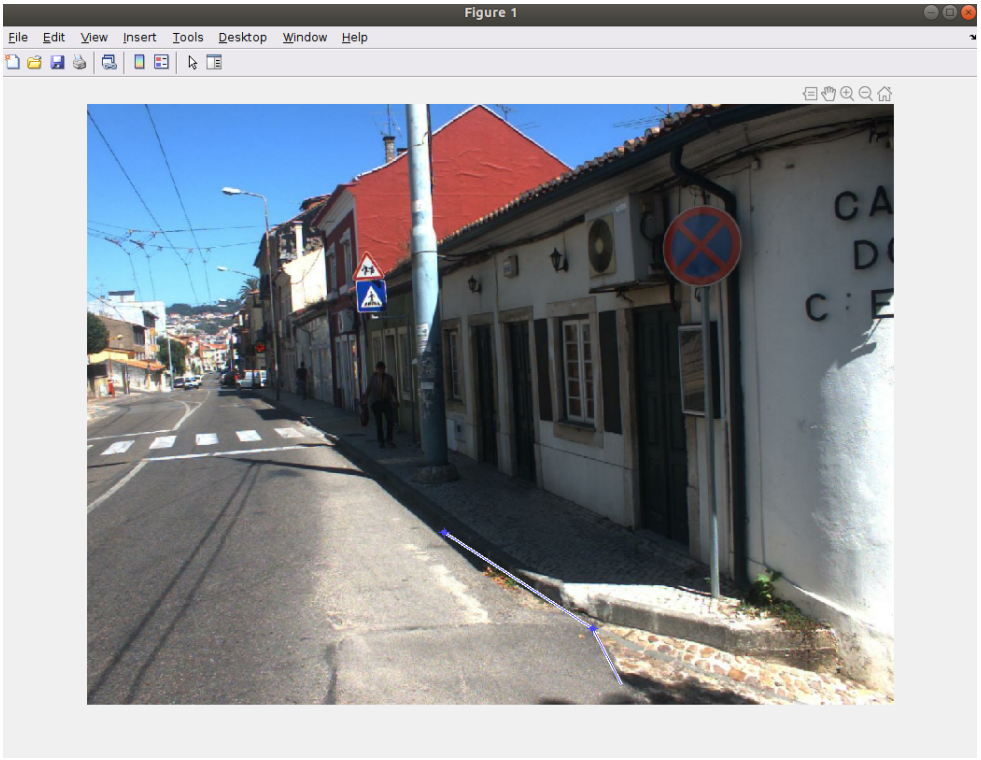

Figure 4.2: Algorithm for generating PPR data. Drawing points in the image.

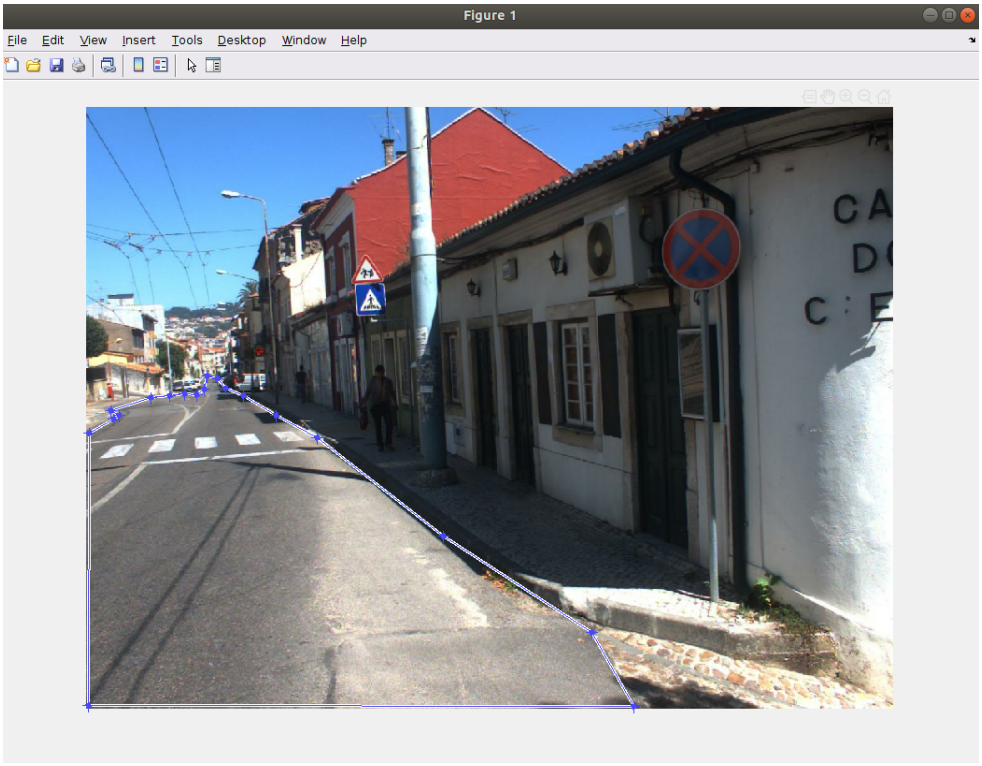

Figure 4.3: Algorithm for generating PPR data. Complete and unfilled drawn area.

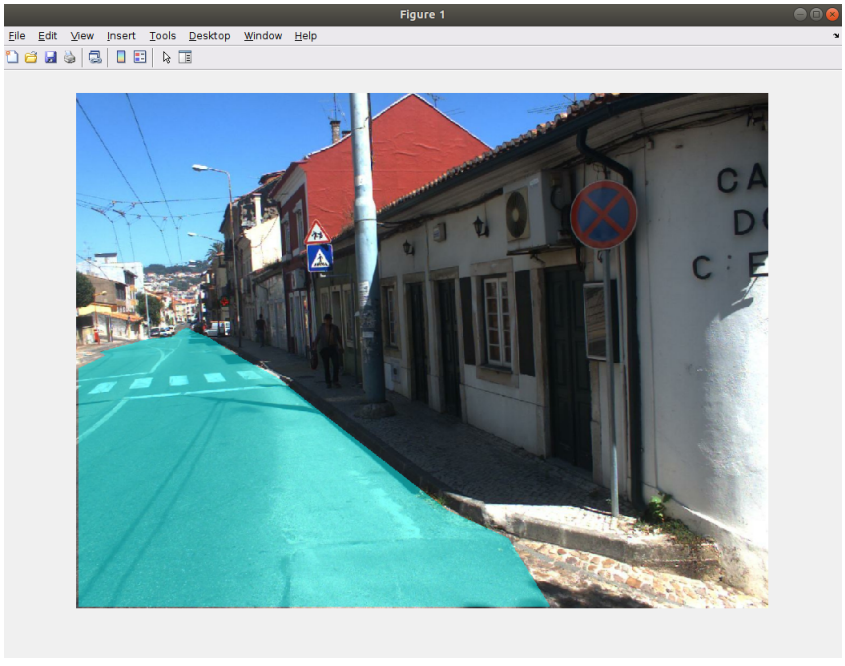

Figure 4.4: Algorithm for generating PPR data. Complete and filled drawn area.

After labeling one area the user can add plane segments to the same plane or define other planes.

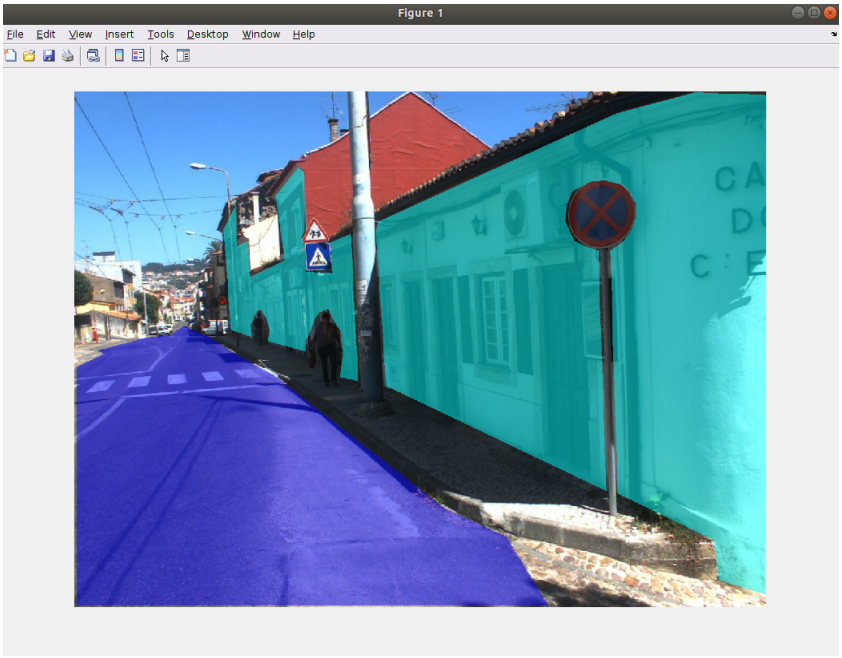

Figure 4.5: Algorithm for generating PPR data. 2 planes identified in an image. Different colors identify different planes.

A final labeled image should resemble something like this:

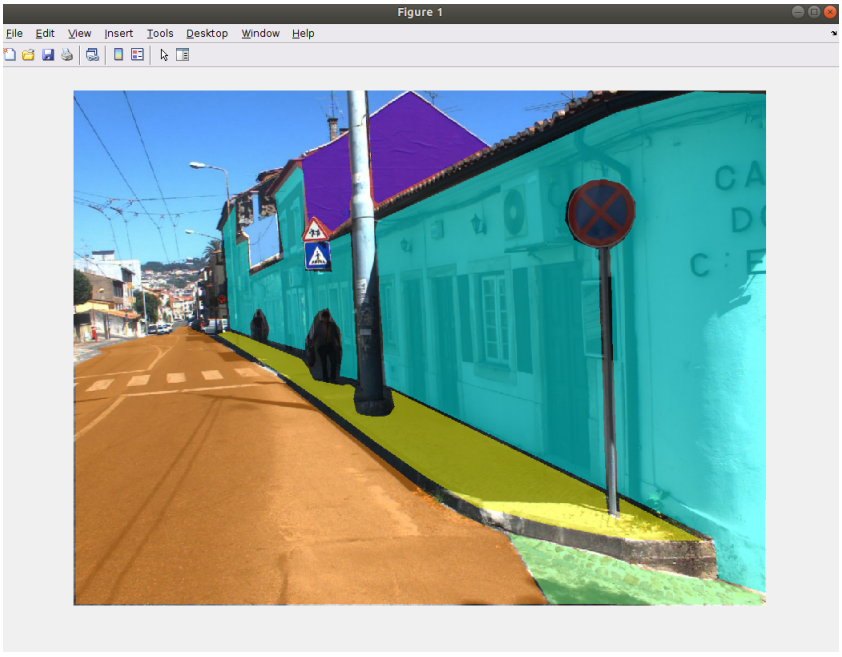

Figure 4.6: Algorithm for generating PPR data. Fully labeled image. Different colors identify different planes.

For each plane labeled, a binary mask is created. A binary mask is an image with just 1 and 0 pixel values. For each binary mask, and using depth and normal maps, is made a plane hypothesis using a RANSAC. RANSAC, as previously mentioned, is an iterative method for estimating pa-

rameters of a mathematical model. The more iterations it makes, the better the chances of getting a good estimation. After obtaining the plane parameters, using the binary mask created for each plane, is found the position of the pixels with value 1. In those locations, in a image filled with 0 with the same size as the original image, the pixel value is changed to a number that corresponds to the order in which the plane was identified. For example, in the location where the second plane was identified, the pixels values will have the value 2. In the end, after labeling all the planes in the image, the code outputs a image where the non-planar structures have the pixel value 0, the first plane detected by the user have the pixel value 1, the second plane the pixel value 2, and so on up to the last plane identified in the image. the information is then saved in $.mat$ file ready to be read by the last part of the algorithm.

In the last part of the algorithm, the user can choose the number of images to which it's done the propagation of the manual planar segmentation. The default value is 10, this means that it's propagated the manual planar segmentation to 5 images sequentially in front of the already labeled image and 4 behind.

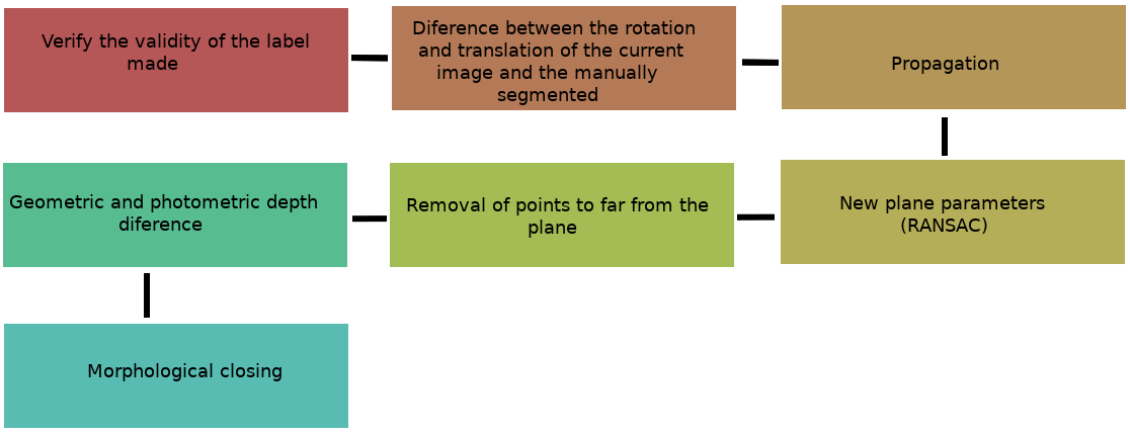

Figure 4.7: The different tasks of the last part of the algorithm.

The first task will verify the validity of the label made. In the area identified by the plane, if the pixels find themselves too far from the camera, they are eliminated from the plane. If the plane is reduced to less than 60% of the original size, the plane is eliminated. The stated percentage may be modified and optimized according to the user's needs.

In the next step, for each image, it's calculated the difference between the pose parameters of the image to which is wanted to propagate the segmentation and the manually segmented image. This way is known where the camera of the propagated image is in relation to the manually segmented image. This information in then used to predict where the points in the area defined by the planes will be in the new image. With the new points and depth and normal maps, it's determined the new plane parameters using RANSAC.

The last step comprises a set of propagated label cleanup processes. The first process checks how far each plane is from its 3D points. If the point is too far from the plane, it is removed from the plane. The second process calculates the difference between the geometric and photometric depth

map information given by the Colmap. If this difference is high, we remove the label assigned to the pixel and set it to 0.Before getting the final planar segmentation image, is used morphological closing to fill gaps in each plane label of the image.

In the end the algorithm outputs in different folders, the propagated planar segmentation image and the plane parameters.

4.2 Proposed PPR dataset

In this dissertation was used several datasets created with images from different areas of the city of Coimbra (Portugal). The datasets were obtained through stereo cameras mounted on the top of a car with well known intrinsics parameters and model. On three of the four datasets used, the car's arrival point was the same as the car's starting point. The datasets feature 1631 images (Santa Clara side-view), 907 images (Santa Clara front-view), 1688 images (Sé Velha), and 2889 images (Ladeira do Seminário). Colmap run through the datasets and was obtained the information needed to fed the PPR data generation agorithm. Every ten images was performed label manual reducing $\frac{1}{10}$ the need for manually label all the images. The algorithm propagates the plane segmentation mask and obtain the plane parameters segmented for 10 images, five in front of the manually label image and four behind. The Figures 4.8, 4.9, 4.10, 4.11, and 4.12 show some results:

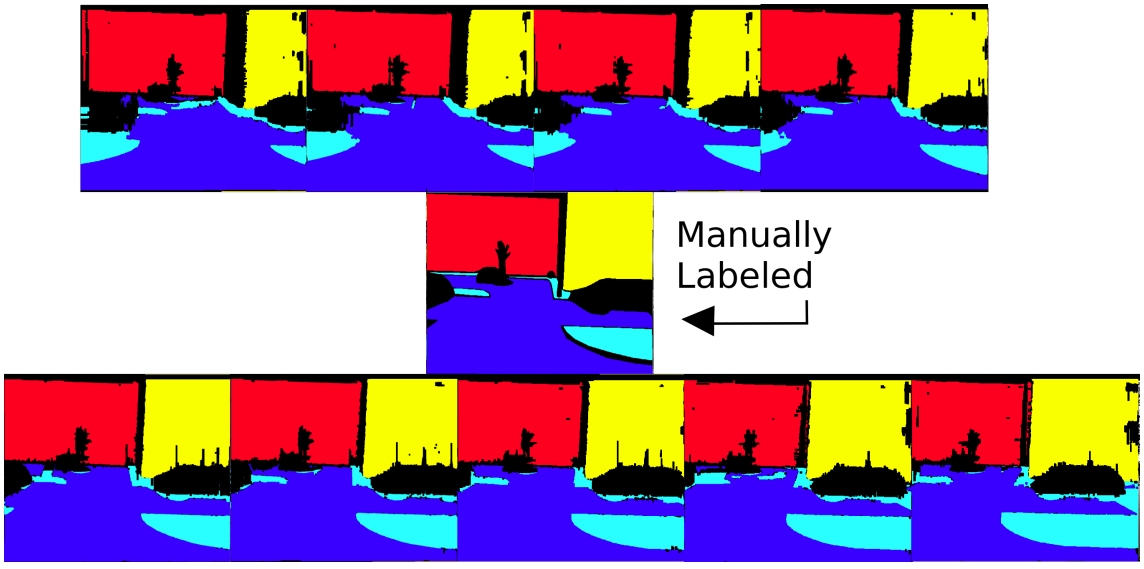

Figure 4.8: Example of a propagation made with one manually labeled image.

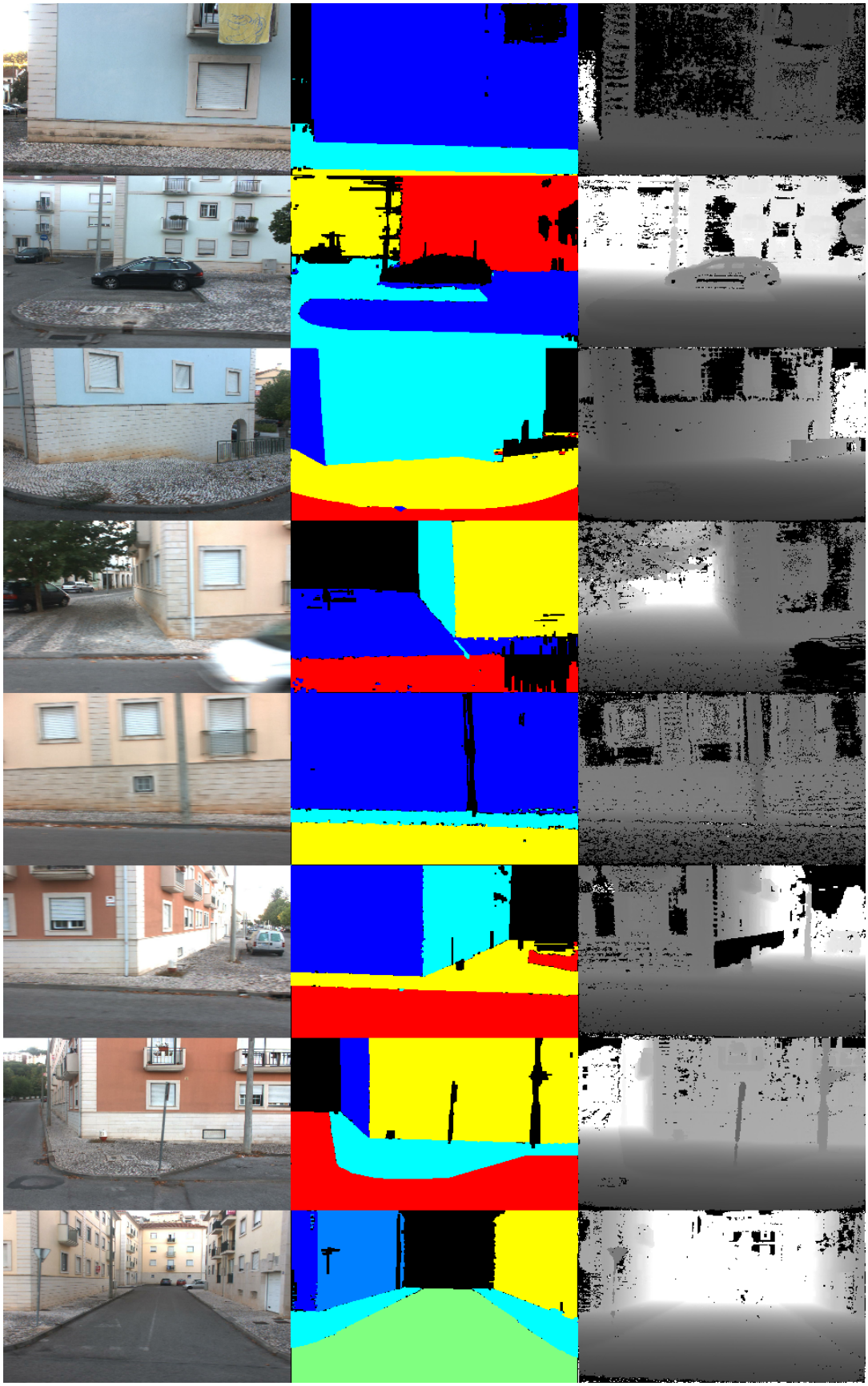

Figure 4.9: Some good labeled and propagated data generated from the Santa Clara side-view by the algorithm created.

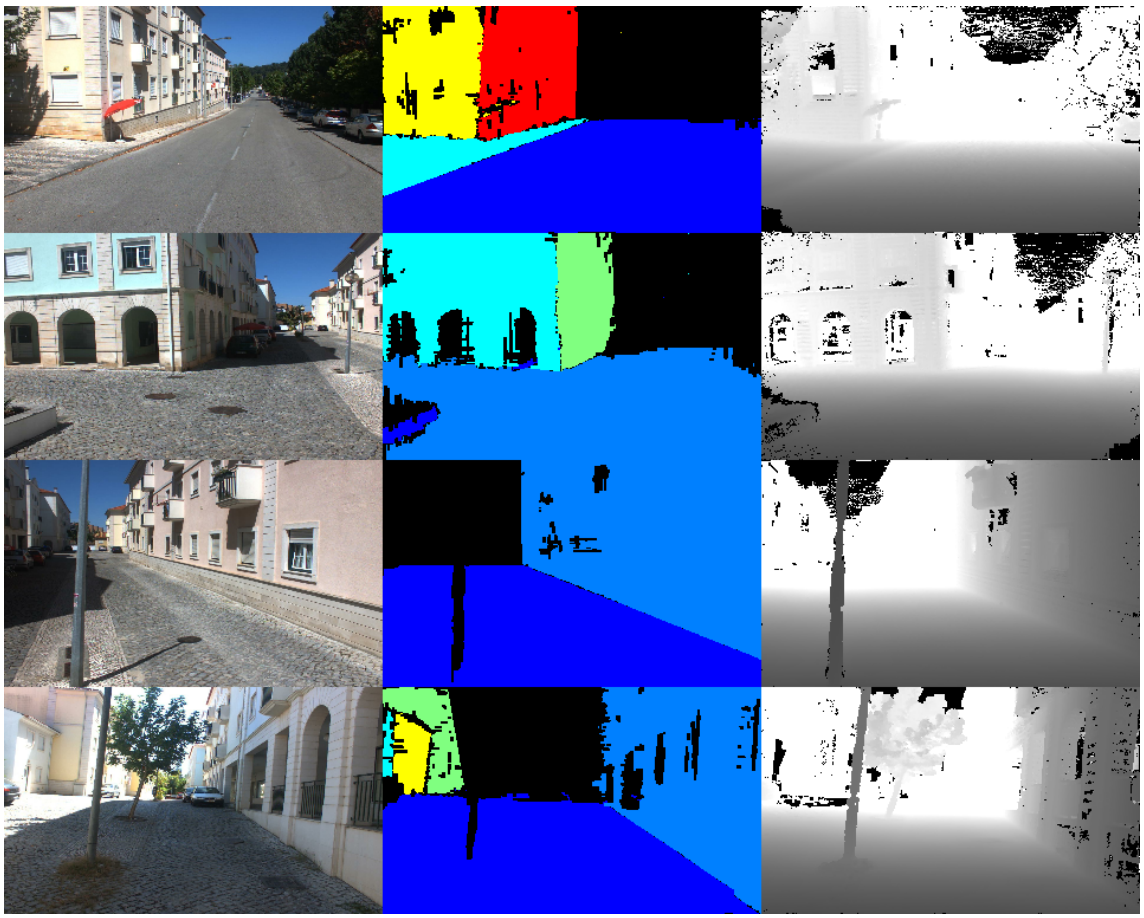

Figure 4.10: Some good labeled and propagated data generated from the Santa Clara front-view by the algorithm created.

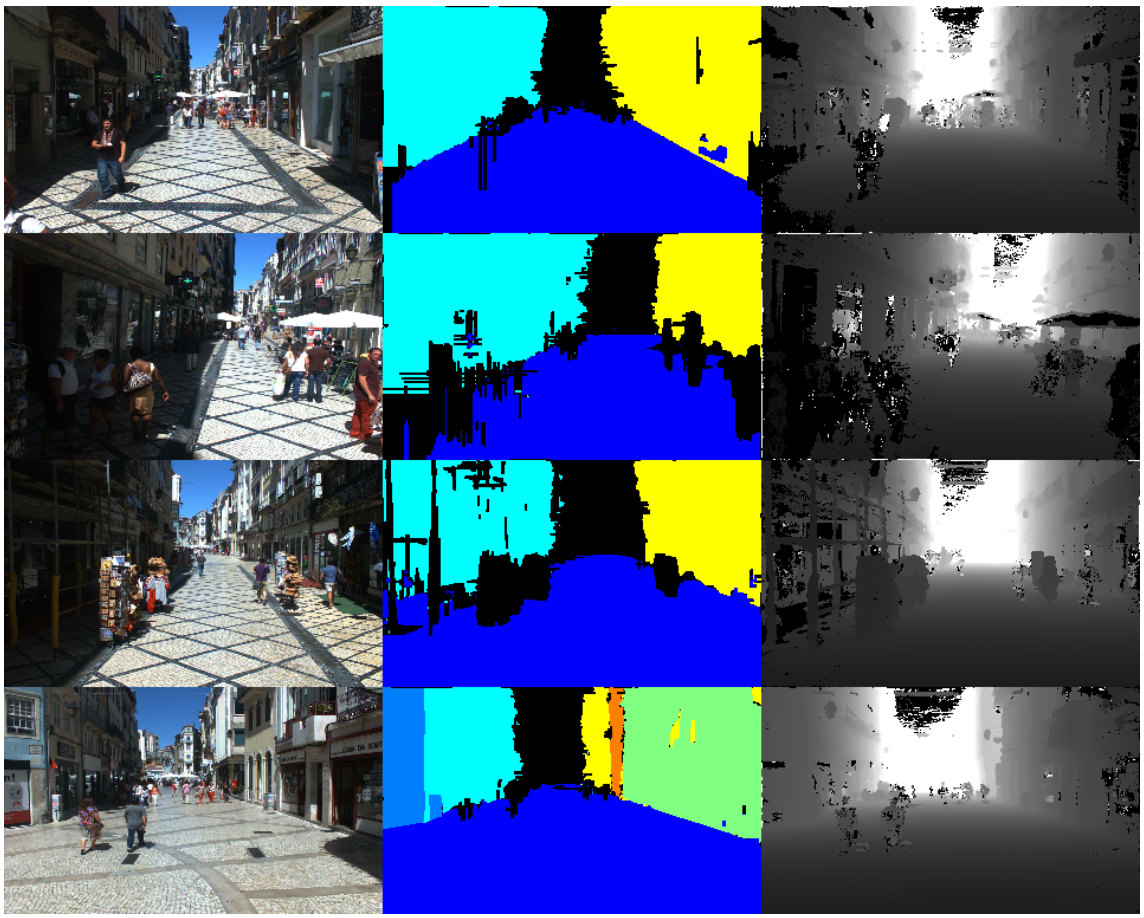

Figure 4.11: Some good labeled and propagated data generated from the Sé Velha by the algorithm created.

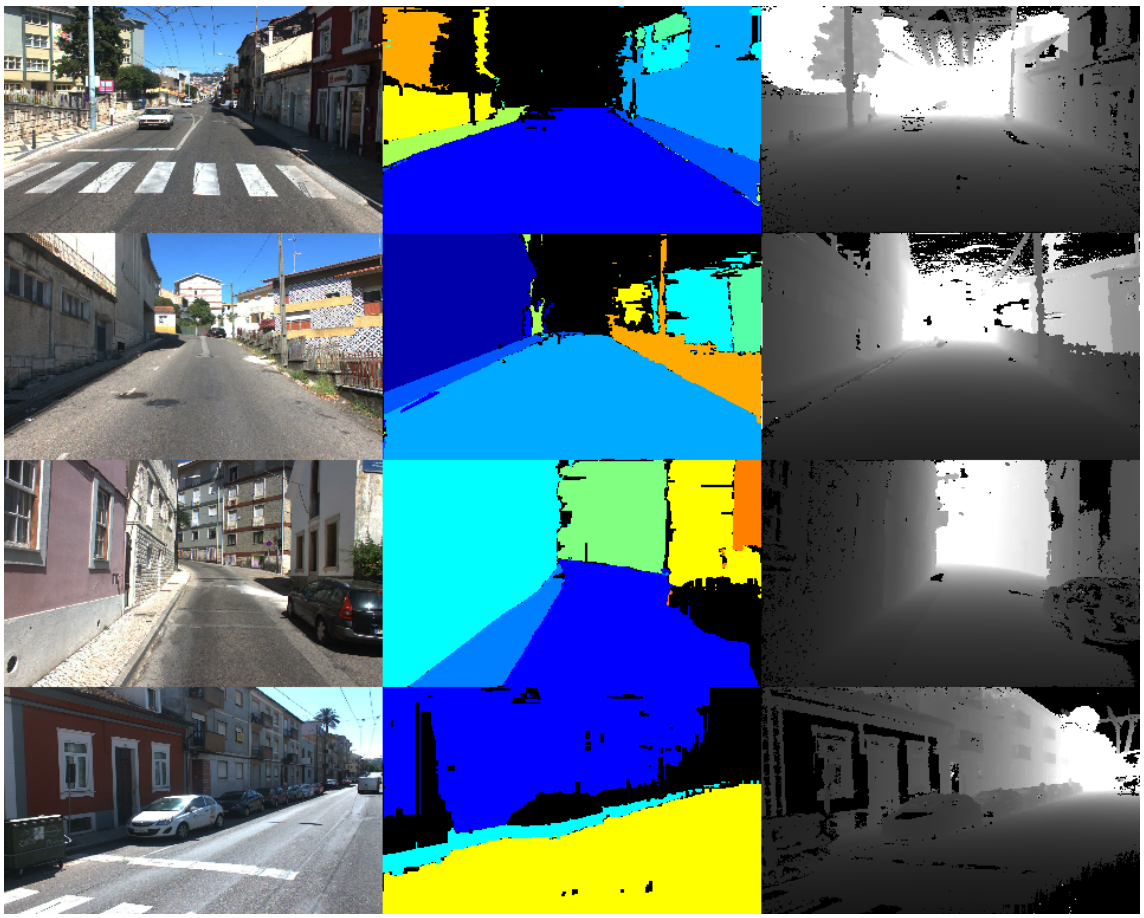

Figure 4.12: Some good labeled and propagated data generated from the Ladeira do Seminário by the algorithm created.

In the images above, from left to right, can seen the original image, the planar segmentation resulting from the manual label propagation performed by the algorithm created, and the depth map generated by Colmap. Looking at the image of the segmented planes (center image), can be concluded that the overall results are good. Images resulting from manual label propagation feature precise segmentation, essential for training PPR deep learning networks. However, the algorithm is not perfect. Occasionally it has some flaws as can be seen in the Figure 4.13.

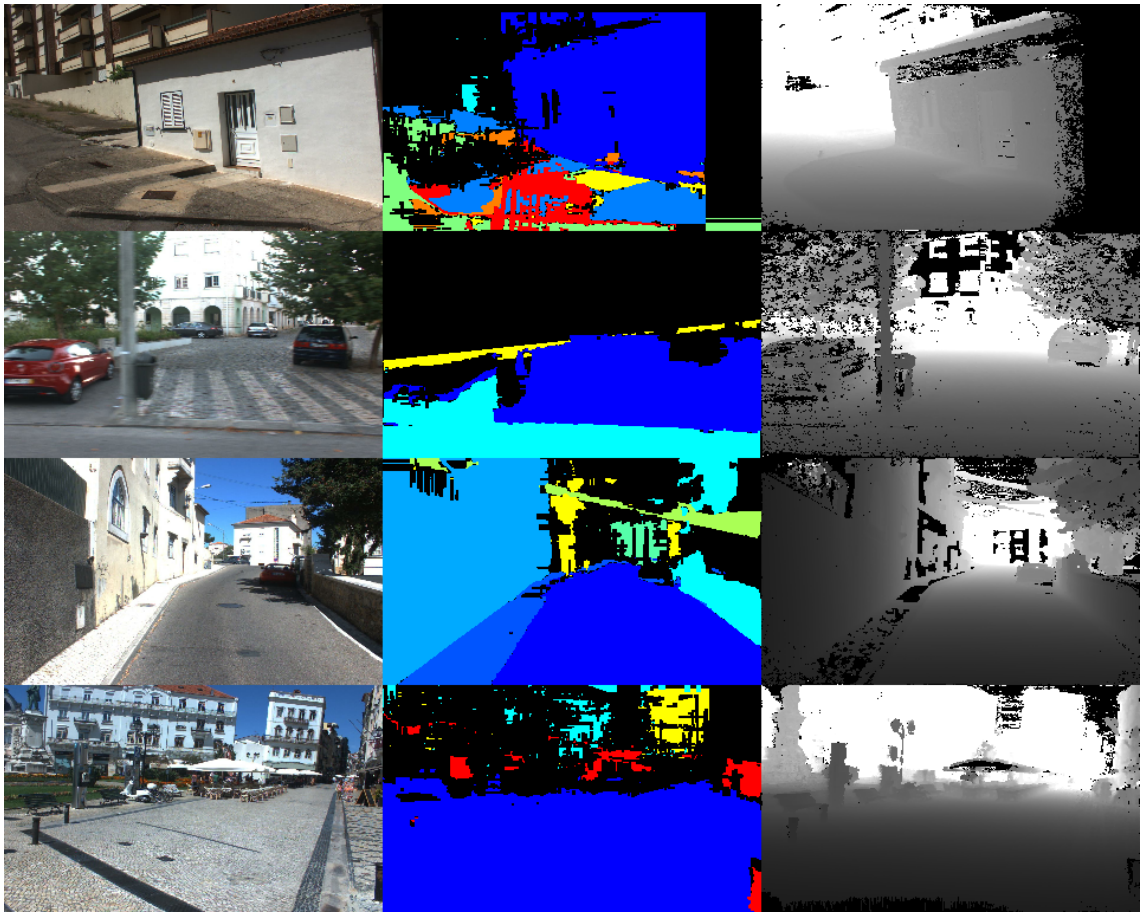

Figure 4.13: Some bad labeled and propagated data generated by the algorithm created.

Here, can easily be seen some errors in the propagation. Sometimes planes are propagated to areas that make no sense. These errors occur due to inaccurate image pose reconstruction and some poor tuning of any of the parameters from the propagation code.

Still, the algorithm present a good solution to the problems involved in obtaining data for deep learning networks that do PPR, compared to the known methods. The algorithm is semi-automatic, fast and shows good results. If even more accurate results need to be obtained, the user can increase the number of RANSAC iterations, label more manual images by propagating to fewer images (reducing propagation errors) and optimizing the tuning parameters of the code.

In the Figure 4.14 can be seen the 3D reconstruction of the planes obtained with the PPR data generation algorithm.

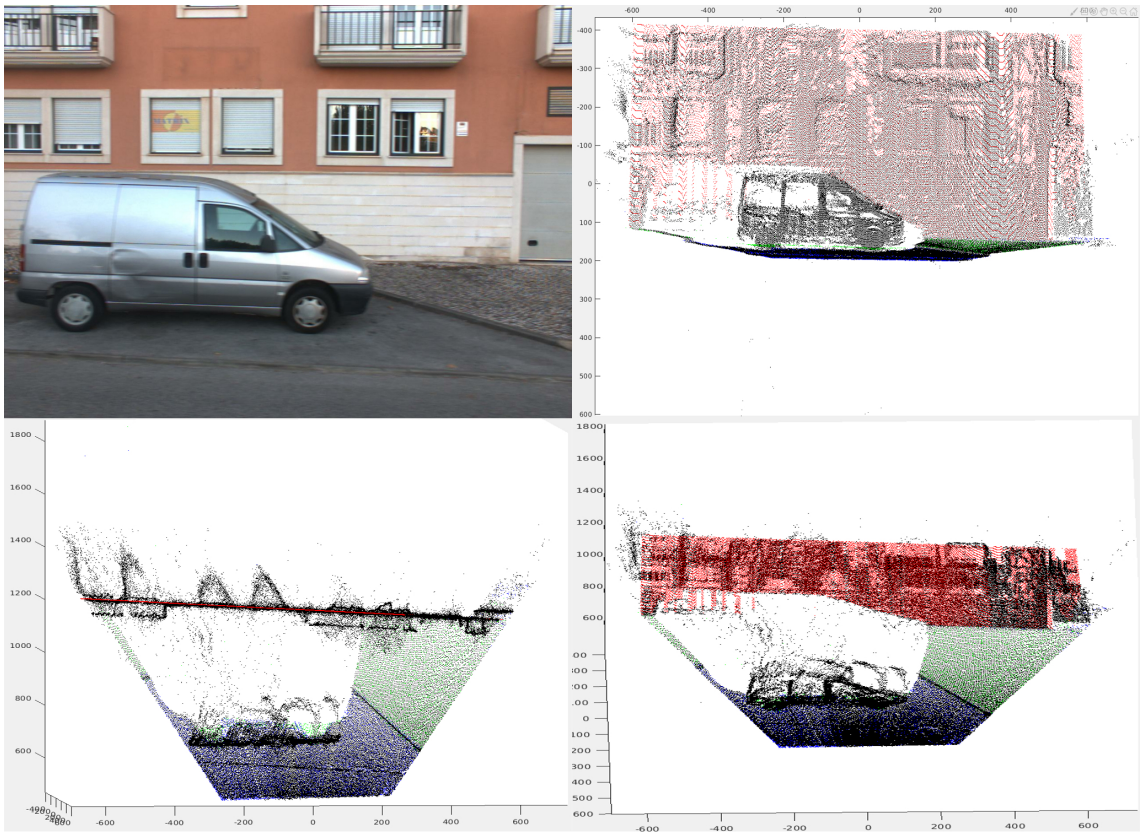

Figure 4.14: 3D reconstruction using the plane parameters given by the PPR data generation algorithm.

5

Single image plane prediction

5.1 Re-training of SI-PPR using proposed dataset

R3PSICNN [27], retrieves planar primitives through a single image. By treating the 3D plane recovery problem as a depth prediction problem, the network was trained using a set of photo-realistic synthetic images of urban scenes with the corresponding depth maps noise free. However, the network has some difficulty correctly distinguish planes in an image of a real scene, as can be seen in the 5.2.To validate the accuracy of the data generated by the algorithm created, R3PSICNN was re-trained.

To be able to re-train the network the user must first create a folder with the training images, their depth map, the binary map that identifies, in the image, planar (with a pixel value of 1) and non-planar structures and a text file with the intrinsic parameters of the image. The depth map results from the output of Colmap's dense reconstruction, which is read from Matlab and written in $uint16$ format. For the binary map, the user must replace all pixel values greater than 1 to 1.

It is also needed a text file named $train\_8000.txt$ that stores the name of the folder where the dataset is stored and, followed by a space, the name of the dataset images to use for training. For example, if the training dataset images are in a folder named 00 and the first image from the dataset is named $000001.jpg$, then the first line of text file will be 00 000001.

Was manually labeled 140 images (80 from Santa Clara side-view, 20 from Santa Clara front-view, Sé Velha, and Ladeira do Seminário), which resulted in about 1260 images for training (720 images from Santa Clara side-view, 180 images from Santa Clara front-view, Sé Velha, and Ladeira do Seminário) and about 140 images for testing (80 images from Santa Clara side-view, 20 images from Santa Clara front-view, Sé Velha, and Ladeira do Seminário).

Before training the network, some changes were made to the test code. The original code calculated the following depth metrics:

$$ABS\_REL = \frac{\sum \frac{|D\_Gt - D\_P|}{D\_Gt}}{n}$$

In this equation, D_Gt corresponds to the ground-truth depth, D_P corresponds to the estimated

depth, and n the number of planes in each image. The lower the value of these metrics, the closer the estimated depth is to ground truth. To further analyze the network re-training result, was added more metrics to their test code. One of the metrics added was a metric that analyzed the difference in angle between the estimated plane and the ground-truth plane:

$$OrienError = acos(AngGt \cdot AngP)$$

This equation calculates the cosine arc of the scalar product between the normalized normals of the ground-truth planes, $AngGt$, and the estimated planes, $AngP$. The lower the value from this metric, the better the result. The value comes in radians. Another metric added was the DICE, Sørensen – Dice coefficient [26, 5]. This metric serves to evaluate the similarity between two samples. In this case, to evaluate the similarity between the ground-truth plane label image and the estimated plane label image:

$$DICE = \frac{2TP}{2TP + FP + FN}$$

For this metric, in the code, is read the estimated and ground-truth plane segmentation images and perform the one-hot encoding operation. One-hot encoding allows the creation of binary masks for each pixel value. For example, in an image, can be found about 5 planes. The segmentation mask is an image wherein non-planar structures the pixel value is 0, the identified first plane has a pixel value of 1, the identified second plane has a pixel value of 2, and so on for the rest of the planes of the image. When applying one-hot encoding, for each pixel value, an image with values 0 and 1 is created. The first image will correspond to the non planar zones. Pixel values 0 will correspond to planar zones, and pixel values 1 will correspond to nonplanar zones. The second image will correspond to the zones categorized as plane 1. Pixels will display values 1 if they are pixels that are in the zone corresponding to plane 1, and value 0 otherwise. The third image will correspond to the zones categorized as plane 2. The pixels will display values 1 if they are pixels that are in the zone corresponding to plane 2, and value 0 otherwise. Similarly for the rest of the one-hot encoding output images. The metric, with the values of each binary mask that exits one-hot encoding, does the calculation of the above equation. TP corresponds to True Positives, FN to False Negatives and FP to False Positives.

In order for the algorithm to calculate the already existing and the added metrics, it is needed to provide the code some ground-truth data. Is needed an imagem, a depth map of the image, a planar segmentation image, a text file with the intrinsic parameters of the image, and a text file with the plane parameters of the planes identified. The depth map results from the output of Colmap's dense reconstruction, which is read from Matlab and written in $uint16$ format. The label image comes from my algorithm.

It is also needed a text file named $tst\_100.txt$ that stores the name of the folder where the dataset is stored and, followed by a space, the name testing dataset images. For example, if the testing

dataset images are in a folder named 00 and the first image from the dataset is named $000001.jpg$, then the first line of text file will be $00\ 000001$.

Taking into account the fact that the network outputs 5 planes, and sometimes the user can label more or less than 5 planes per image, was created a system to sort the estimated planes to the ground-truth planes. The sorting performed is based on the DICE metric. For a given ground-truth plane, the DICE metric is calculated for all estimated planes. The one that shows the best result will be given the respective ground-truth plane to calculate all the other metrics. The correspondence between ground-truth and estimate is unique, the same estimated plane cannot be compared with two or more ground-truth planes and vice versa. The Figure 5.1, shows visually the sorting implemented in the code.

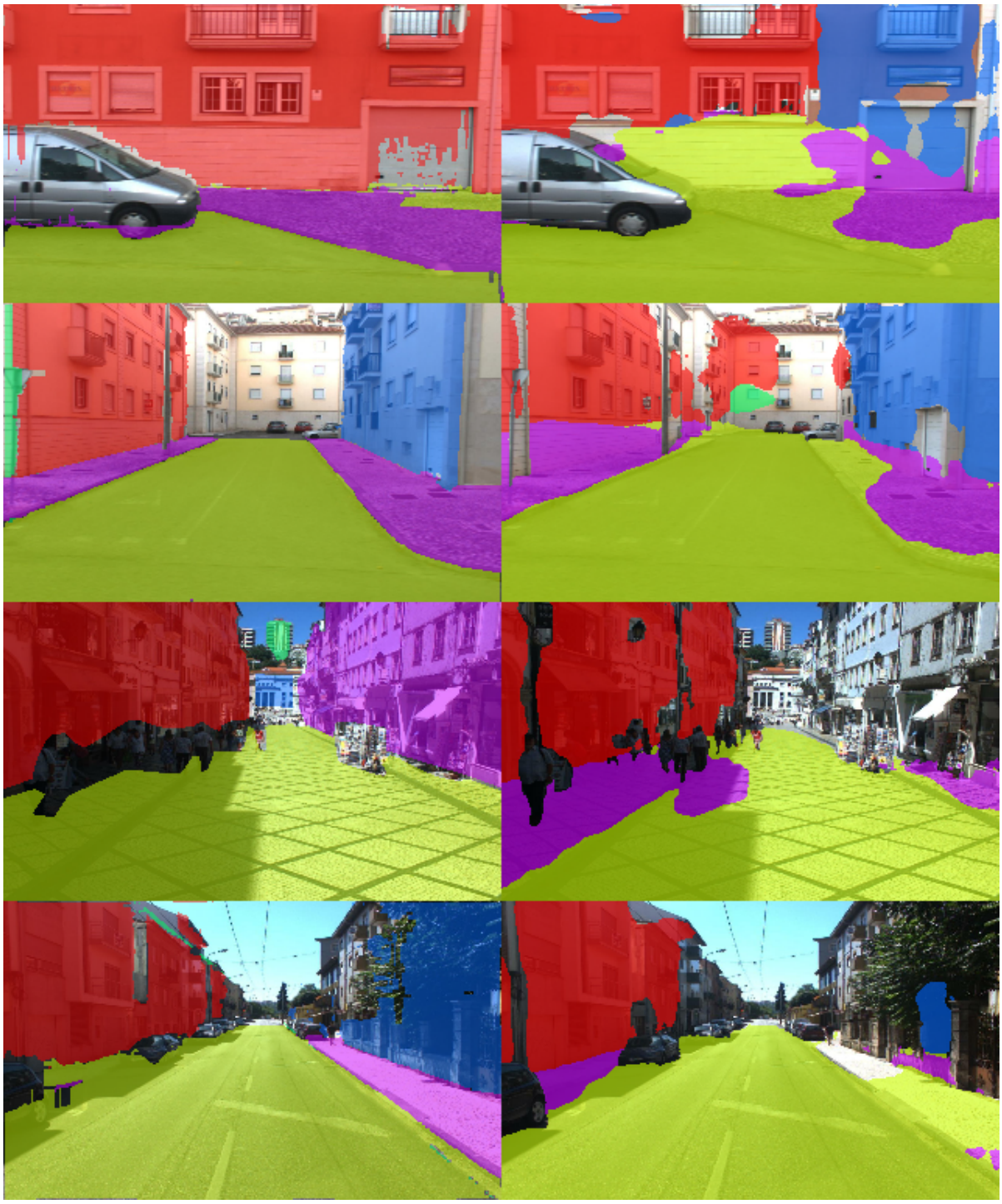

Figure 5.1: Visually representation of the sorting system made. The groud-truth is in the left and the output from the algorithm before re-training is in the right. Planes with the same color are the planes to be compared with each other.

5.2 Experimental results

Here, it will be presented some results. The following table shows the values of the metrics described in the previous section before re-training, using a pre-trained model of the deep learning algorithm R3PSICNN to estimate planar structures, and after re-training with the PPR data generated by the algorithm:

	ABS_REL	OrienError	DICE
Before re-training	0.37822196	0.4368921	47.7%
After re-training	**0.22695127**	**0.3337691**	**51.1%**

Table 5.1: Metric results in the testing dataset, before and after re-training the algorithm with the training dataset.

As can be seen from the table presented, after re-training the network with the PPR data generated by the algorithm, there are improvements in the metrics. It can be concluded that the PPR data generated is accurate enough to train efficiently deep learning algorithms and leverage the network capability of plane segmentation in real scenes images.

The Figure 5.2 shows the planes segmentation output from the piece-wise planar deep learning algorithm used before the re-training, after the re-training and the ground-truth, respectively:

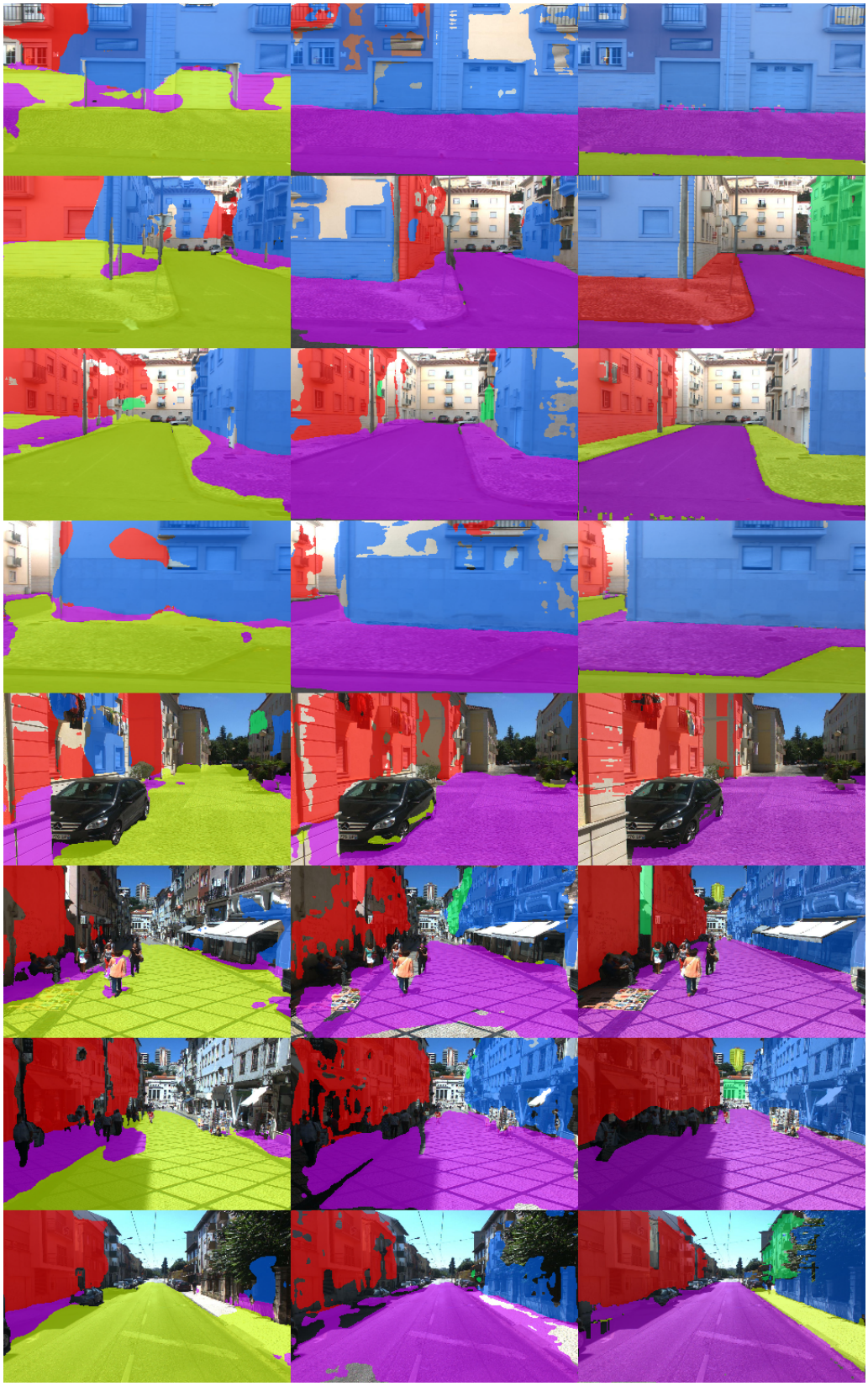

Figure 5.2: Testing of the PPR deep learning algorithm in some images from the test dataset. From left to right we have the plane segmentation before the re-training, the plane segmentation after the re-training and the ground-truth.

Visually, can be seen an improvement in the plane segmentation after re-training the algorithm as expected due to the results of the metrics used. The planes are more well defined and in the right location compared to the plane segmentation images from the algorithm before re-training. However there were also bad results (Figure 5.3). In the figure can be seen different planes labeled as the same and two or more planes identified in areas where there is only one plane.

Figure 5.3: Some failure cases from the test dataset.

6

Conclusion

Deep learning algorithms require a large amount of accurate and varied data to achieve good results. DL algorithms that attempt to identify planar structures in an image are no exception. Obtaining this type of information accurately and in large quantities is difficult. There are problems with getting the right hardware for the purpose at hand, the difficulty associated with operating these types of equipment and the time required to do so, and even the time needed to manually segment planes in every image. The pipeline created to generate PPR data can solve these problems by presenting a solution that can replace the need for expensive hardware and considerably reduce the time needed to acquire such information. As it was shown in the previous chapters, the generated data can effectively train the networks to which the data is intended. However, there is work that can still be developed to improve the spread of manual segmentation. To improve results, it is possible to optimize the parameters related to feature extraction and their matching between sequential images in Colmap. Thus, errors associated with the poor pose estimation of the images are prevented and the propagation is better. It is also possible to pay more attention to the propagation itself. Results can be improved by changing the number of iterations performed during RANSAC, increasing the number of images to be manually segmented and reducing the number of images to propagate the manual segmentation, and even tuning the parameters involved in the pipeline in order to optimize the data generation for each dataset used. However, it must be borne in mind that modifying these parameters can reduced the speed in which the data is generated. To conclude, in this dissertation is presented the first pipeline that is able to efficiently and accurately generate PPR data from real images. The pipeline is composed by three main stages: (1) depth data generation using colmap, (2) manual labeling of a small percentage of the images of the dataset, and (3) automatic label and plane propagation to neighboring views using a new strategy based on geometric constraints and random sampling. As future work, we aim to investigate three aspects: (1) use of image collections available in the internet, (2) re-train other SIDE-PPR networks and experimentally testing them, and (3) investigate new strategies for obtaining a complete automatic PPR data generation pipeline.

Bibliography

[1] M. Antunes, J. P. Barreto, and U. Nunes. Piecewise-planar reconstruction using two views. *Image and Vision Computing*, 2016.

[2] C.Godard, O.M.Aodha, and G.J.Brostow. Unsupervised monocular depth estimation with left-right consistency. *CVPR*, 2017.

[3] C.Raposo and J.Barreto. π-match: Monocular vslam and piecewise planar reconstruction using fast plane correspondences. *European Conference on Computer Vision*, 2016.

[4] C.Raposo, M.Antunes, and J.Barreto. Piecewise-planar stereoscan:structure and motion from plane primitives. *ECCV*, 2014.

[5] Lee R Dice. Measures of the amount of ecologic association between species. *Ecology. 26 (3): 297–302*, 1945.

[6] D.Scharstein and R. Szeliski. A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. *International Journal of Computer Vision*, 2002.

[7] David Eigen and Rob Fergus. Predicting depth, surface normals and semantic labels with a common multi-scale convolutional architecture. *In Proceedings of the IEEE International Conference on Computer Vision, pages 2650–2658, 2015*, doi: 10.1109/ICCV.2015.304.

[8] Gallup et al. Piecewise planar and non-planar stereo for urban scene reconstruction. *CVPR*, 2010.

[9] Liu et al. Planenet: Piece-wise planar reconstruction from a single rgb image. *CVPR*, 2018.

[10] Paolo Favaro and Stefano Soatto. A geometric approach to shape from defocus. *IEEE*, 2005.

[11] Guosheng Lin Fayao Liu, Chunhua Shen and Ian Reid. Learning depth from single monocular images using deep convolutional neural fields. *IEEE transactions on pattern analysis and machine intelligence*, 38(10):2024–2039, 2016.

[12] D. Gallup, J.-M. Frahm, and M. Pollefeys. Piecewise planar and non-planar stereo for urban scene reconstruction. *CVPR*, 2010.

[13] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. Deep learning. *MIT Press*, 2016.

[14] Richard Hartley and Andrew Zisserman. Multiple view geometry in computer vision. *Cambridge University Press New York, NY, USA* ©, 2003.

[15] Berthold K. P. Horn. A method for obtaining the shape of a smooth opaque object from oneview. *Technical report, MIT, Cambridge, MA, USA*, 1970.

[16] I.Laina, C.Rupprecht, V.Belagiannis, F.Tombaril, and N.Navab. Deeper depth prediction with fully convolutional residual networks. *IEEE 3DV*, 2016.

[17] Isack and Boykov. Energy-based geometric multi-model fitting. *IJCV*, 2012.

[18] T. Koch, L. Liebel, F. Fraundorfer, and M.Körner. Evaluation of cnn-based single-image depth estimation methods. *arXiv:1805.01328*, 2018.

[19] D. Lowe. Distinctive image features from scale-invariant keypoints. *IJCV*, 2004.

[20] Yi Ma, Jana Kosecka Stefano Soatto, S. Shankar Sastry, and SpringerVerlag. An invitation to 3-d vision. ©, 2003.

[21] Sumit Saha.

[22] Johannes Lutz Schönberger and Jan-Michael Frahm. Structure-from-motion revisited. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[23] Johannes Lutz Schönberger, True Price, Torsten Sattler, Jan-Michael Frahm, and Marc Pollefeys. A vote-and-verify strategy for fast spatial verification in image retrieval. In *Asian Conference on Computer Vision (ACCV)*, 2016.

[24] Johannes Lutz Schönberger, Enliang Zheng, Marc Pollefeys, and Jan-Michael Frahm. Pixelwise view selection for unstructured multi-view stereo. In *European Conference on Computer Vision (ECCV)*, 2016.

[25] S. Seitz, B. Curless, J. Diebel, D. Scharstein, and R. Szelisk. A comparison and evaluation of multi-view stereo reconstruction algorithms. *CVPR*, 2016.

[26] T Sørensen. A method of establishing groups of equal amplitude in plant sociology based on similarity of species and its application to analyses of the vegetation on danish commons. *Kongelige Danske Videnskabernes Selskab. 5 (4): 1–34*, 1948.

[27] Fengting Yang and Zihan Zhou. Recovering 3d planes from a single image via convolutional neural networks. *ECCV*, 2018.

[28] F. Yu, V. Koltun, and T. Funkhouser. Dilated residual networks. *arXiv preprint arXiv:1705.09914*, 2017.

[29] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia. Pyramid scene parsing network. *arXiv preprint arXiv:1612.01105*, 2016.

[30] S. Zheng, S. Jayasumana, B. Romera-Paredes, V. Vineet, Z. Su, D. Du, C. Huang, and P. H. Torr. Conditional random fields as recurrent neural networks. in proceedings of the ieee international conference on computer vision. *pages 1529–1537*, 2015.

[31] Noah Snavely Zhengqi Li. Megadepth: Learning single-view depth prediction from internet photos. *CVPR*, 2018.